



```
DDDDDDDD  BBBB BBBB  GGGGGGGG  SSSSSSSS  YY  YY  MM  MM  BBBB BBBB  000000  LL
DDDDDDDD  BBBB BBBB  GGGGGGGG  SSSSSSSS  YY  YY  MM  MM  BBBB BBBB  000000  LL
DD  DD  BB  BB  GG  SS  YY  YY  MMMM  MMMM  BB  BB  00  00  LL
DD  DD  BB  BB  GG  SS  YY  YY  MMMM  MMMM  BB  BB  00  00  LL
DD  DD  BB  BB  GG  SS  YY  YY  MM  MM  BB  BB  00  00  LL
DD  DD  BBBB BBBB  GG  GG  YY  YY  MM  MM  BBBB BBBB  00  00  LL
DD  DD  BBBB BBBB  GG  GG  YY  YY  MM  MM  BBBB BBBB  00  00  LL
DD  DD  BB  BB  GG  GG  GG  YY  MM  MM  BB  BB  00  00  LL
DD  DD  BB  BB  GG  GG  GG  YY  MM  MM  BB  BB  00  00  LL
DD  DD  BB  BB  GG  GG  GG  YY  MM  MM  BB  BB  00  00  LL
DD  DD  BB  BB  GG  GG  GG  YY  MM  MM  BB  BB  00  00  LL
DDDDDDDD  BBBB BBBB  GGGGGG  SSSSSSSS  YY  MM  MM  BBBB BBBB  000000  LLLLLLLLLL
DDDDDDDD  BBBB BBBB  GGGGGG  SSSSSSSS  YY  MM  MM  BBBB BBBB  000000  LLLLLLLLLL

LL  I I I I I  SSSSSSSS
LL  I I I I I  SSSSSSSS
LL  I I  SS
LL  I I  SS
LL  I I  SS
LL  I I  SS
LL  I I  SSSSSS
LL  I I  SSSSSS
LL  I I  SS
LL  I I  SS
LL  I I  SS
LL  I I  SS
LLLLLLLLLL  I I I I I  SSSSSSSS
LLLLLLLLLL  I I I I I  SSSSSSSS
```



```
1 0001 0 MODULE DBGSYMBOL (IDENT = 'V04-000') =
2 0002 0
3 0003 1 BEGIN
4 0004 1
5 0005 1 *****
6 0006 1 *
7 0007 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
8 0008 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
9 0009 1 * ALL RIGHTS RESERVED.
10 0010 1 *
11 0011 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
12 0012 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
13 0013 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
14 0014 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
15 0015 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
16 0016 1 * TRANSFERRED.
17 0017 1 *
18 0018 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
19 0019 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
20 0020 1 * CORPORATION.
21 0021 1 *
22 0022 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
23 0023 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
24 0024 1 *
25 0025 1 *
26 0026 1 *****
27 0027 1
28 0028 1 WRITTEN BY
29 0029 1 Ping Sager March, 1982
30 0030 1
31 0031 1 MODULE FUNCTION
32 0032 1 This module contains all routines connected with finding out
33 0033 1 more information about DEBUG's symbols. It thus implements the
34 0034 1 SHOW SYMBOL, SYMBOLIZE commands. SHOW SYMBOL command lists all
35 0035 1 symbols in DEBUG's Run-Time Symbol Table (the RST) which have a
36 0036 1 given single symbol name (%LABEL n is a symbol name), or a symbol
37 0037 1 name which includes wildcard character (the allowed wildcard
38 0038 1 character is "*", which matches zero or more characters) in a
39 0039 1 specified scope (a module, a routine, or a lexical block).
40 0040 1 Qualifiers on the SHOW SYMBOL command which specify the additional
41 0041 1 information is to be shown. SYMBOLIZE command shows all feasible
42 0042 1 symbolic representation of a given address.
43 0043 1
44 0044 1
45 0045 1 REQUIRE 'SRC$;DBGPROLOG.REQ';
46 0179 1
47 0180 1 FORWARD ROUTINE
48 0181 1 DBG$STA_ADDRSPEC, Interpret a DST value spec
49 0182 1 DBG$STA_SHOWSYMBOL, Handle the SHOW SYMBOL command
50 0183 1 DBG$STA_SYMADDR, Get a symbol's value or address
51 0184 1 representation
52 0185 1 DBG$TRANSLATE_DESCR: NOVALUE, Given pointer to VAX Standard
53 0186 1 Descriptor, translate its
54 0187 1 information into user readable
55 0188 1 form
56 0189 1 DBG$TRANSLATE_KIND, Translate RST's kind field into
57 0190 1 user readable form
```



|    |      |   |                                   |                                         |
|----|------|---|-----------------------------------|-----------------------------------------|
| 58 | 0191 | 1 | DBG\$TRANSLATE_TYPECODE: NOVALUE, | Translate Data Type Code into user      |
| 59 | 0192 | 1 |                                   | readable form                           |
| 60 | 0193 | 1 | DBG\$TRANSLATE_REG: NOVALUE,      | Translate number into register name     |
| 61 | 0194 | 1 | DBG\$DUMP_HEX: NOVALUE,           | Given the address, dump the specified   |
| 62 | 0195 | 1 |                                   | number of bytes in hex                  |
| 63 | 0196 | 1 | BLD_SCOPE_SEARCH_LIST: NOVALUE,   | Allocate memory block for scope search  |
| 64 | 0197 | 1 |                                   | list or expand this list and            |
| 65 | 0198 | 1 |                                   | enter the scope to the list             |
| 66 | 0199 | 1 | GET_BLISS_SPECIAL_CASES: NOVALUE, | Get information for the Bliss Special   |
| 67 | 0200 | 1 |                                   | Case DST Record                         |
| 68 | 0201 | 1 | GET_VARIANT: NOVALUE,             | Get information for a variant from      |
| 69 | 0202 | 1 |                                   | RST entry                               |
| 70 | 0203 | 1 | SHOW_SYMBOL: NOVALUE,             | Show the information about a given      |
| 71 | 0204 | 1 |                                   | symbol                                  |
| 72 | 0205 | 1 | SHOW_SYMBOL_ADDRESS: NOVALUE,     | Show symbol's address info for /ADDRESS |
| 73 | 0206 | 1 | SHOW_SYMBOL_TYPE: NOVALUE,        | Show symbol's type info for /TYPE       |
| 74 | 0207 | 1 | SHOW_SYMBOL_TYPE_HANDLER,         | Handles the abnormal conditions occur   |
| 75 | 0208 | 1 |                                   | during the DST Value Spec               |
| 76 | 0209 | 1 |                                   | evaluation                              |
| 77 | 0210 | 1 | WILDCARD_CANDIDATES,              | Handle symbol name has wildcard format  |
| 78 | 0211 | 1 | WILDCARD_NAME_SETUP,              | Set up wildcard symbol name for string  |
| 79 | 0212 | 1 |                                   | match processing                        |
| 80 | 0213 | 1 | WILDCARD_SUB_STRING;              | Break wildcard symbol name into sub-    |
| 81 | 0214 | 1 |                                   | strings                                 |



|     |      |   |                                     |                                                                                  |
|-----|------|---|-------------------------------------|----------------------------------------------------------------------------------|
| 83  | 0215 | 1 | EXTERNAL ROUTINE                    |                                                                                  |
| 84  | 0216 | 1 | DBG\$GET_DST_NAME,                  | Get the ASCII name from a DST record                                             |
| 85  | 0217 | 1 | DBG\$GET_MEMORY,                    | Get a memory block from memory pool                                              |
| 86  | 0218 | 1 | DBG\$GET_TEMPMEM,                   | Get a temporary memory block from memory pool                                    |
| 87  | 0219 | 1 |                                     |                                                                                  |
| 88  | 0220 | 1 | DBG\$HASH_FIND,                     | Find a name in the RST hash table                                                |
| 89  | 0221 | 1 | DBG\$HASH_FIND_SETUP: NOVALUE,      | Set up calls on HASH_FIND routine                                                |
| 90  | 0222 | 1 | DBG\$LANGUAGE,                      | Produce name of given language                                                   |
| 91  | 0223 | 1 | DBG\$MAKE_INTEGER_DESC,             | Create value descriptor for integer                                              |
| 92  | 0224 | 1 | DBG\$NEWLINE: NOVALUE,              | Output the current print buffer and initialize a new print line                  |
| 93  | 0225 | 1 |                                     |                                                                                  |
| 94  | 0226 | 1 | DBG\$NPATHDESC_TO_CS: NOVALUE,      | Generate pathname ASCII string from a pathname descriptor                        |
| 95  | 0227 | 1 |                                     |                                                                                  |
| 96  | 0228 | 1 | DBG\$POP_TEMPMEM: NOVALUE,          | Pop the pointer to temporary memory blocks off the Temporary Memory Pool Stack   |
| 97  | 0229 | 1 |                                     |                                                                                  |
| 98  | 0230 | 1 |                                     |                                                                                  |
| 99  | 0231 | 1 | DBG\$PRINT: NOVALUE,                | Output text to the print buffer using FAO formatting                             |
| 100 | 0232 | 1 |                                     |                                                                                  |
| 101 | 0233 | 1 | DBG\$PRINT_CONTROL,                 | Control the print indentation levels                                             |
| 102 | 0234 | 1 | DBG\$PUSH_TEMPMEM,                  | Push the pointer to temporary memory blocks onto the Temporary Memory Pool Stack |
| 103 | 0235 | 1 |                                     |                                                                                  |
| 104 | 0236 | 1 |                                     |                                                                                  |
| 105 | 0237 | 1 | DBG\$REL_MEMORY: NOVALUE,           | Release a memory block to memory pool                                            |
| 106 | 0238 | 1 | DBG\$RST_BLDSCOPE_LIST,             | Set up the set scope search list                                                 |
| 107 | 0239 | 1 | DBG\$RST_DST_PTR,                   | Get the 'real' DST pointer                                                       |
| 108 | 0240 | 1 | DBG\$RST_TEMP_RELEASE: NOVALUE,     | Release all temporary RST entries which are not locked                           |
| 109 | 0241 | 1 |                                     |                                                                                  |
| 110 | 0242 | 1 | DBG\$SAVE_VAL: NOVALUE,             | Save ""                                                                          |
| 111 | 0243 | 1 | DBG\$STA_NUMBERED_SCOPE: NOVALUE,   | Convert numbered scope to module RST and scope RST pointers                      |
| 112 | 0244 | 1 |                                     |                                                                                  |
| 113 | 0245 | 1 | DBG\$STA_SETCONTEXT: NOVALUE,       | Set up context for value evaluation                                              |
| 114 | 0246 | 1 | DBG\$STA_SYMKIND: NOVALUE,          | Get a symbol's kind                                                              |
| 115 | 0247 | 1 | DBG\$STA_SYMPATHNAME: NOVALUE,      | Get a symbol's full pathname                                                     |
| 116 | 0248 | 1 | DBG\$STA_SYMTYPE: NOVALUE,          | Return a symbol's TYPEID and FCODE                                               |
| 117 | 0249 | 1 | DBG\$STA_TYP_AREA: NOVALUE,         | Return info for a PL/I area type                                                 |
| 118 | 0250 | 1 | DBG\$STA_TYP_ARRAY: NOVALUE,        | Return info for an array type                                                    |
| 119 | 0251 | 1 | DBG\$STA_TYP_ATOMIC: NOVALUE,       | Return type code for an atomic type                                              |
| 120 | 0252 | 1 | DBG\$STA_TYP_DESCR: NOVALUE,        | Return descriptor for descriptor type                                            |
| 121 | 0253 | 1 | DBG\$STA_TYP_ENUM: NOVALUE,         | Return info for an enumeration type                                              |
| 122 | 0254 | 1 | DBG\$STA_TYP_FILE: NOVALUE,         | Return info for a file type                                                      |
| 123 | 0255 | 1 | DBG\$STA_TYP_OFFSET: NOVALUE,       | Return info for a PL/I offset type                                               |
| 124 | 0256 | 1 | DBG\$STA_TYP_PICT: NOVALUE,         | Return info for a picture type                                                   |
| 125 | 0257 | 1 | DBG\$STA_TYP_TYPEDPTR: NOVALUE,     | Return info for a typed pointer type                                             |
| 126 | 0258 | 1 | DBG\$STA_TYP_RECORD: NOVALUE,       | Return info for a record type                                                    |
| 127 | 0259 | 1 | DBG\$STA_TYP_SET: NOVALUE,          | Return info for a set type                                                       |
| 128 | 0260 | 1 | DBG\$STA_TYP_SUBRNG: NOVALUE,       | Return info for a subrange type                                                  |
| 129 | 0261 | 1 | DBG\$STA_TYP_VARIANT: NOVALUE,      | Return info for a variant type                                                   |
| 130 | 0262 | 1 | DBG\$STA_TYP_VARIANT_COMP: NOVALUE, | XXX                                                                              |
| 131 | 0263 | 1 | OTSSCVT_L_TZ,                       | Converts a string to ASCII hex format                                            |
| 132 | 0264 | 1 |                                     |                                                                                  |
| 133 | 0265 | 1 | STR\$CONCAT,                        | Concatenate two strings                                                          |
| 134 | 0266 | 1 | STR\$DUPL_CHAR,                     | Generates a string containing n duplicates of the input character                |
| 135 | 0267 | 1 |                                     |                                                                                  |
| 136 | 0268 | 1 | STR\$FREE1_DX;                      | Free a dynamic string                                                            |
| 137 | 0269 | 1 |                                     |                                                                                  |
| 138 | 0270 | 1 |                                     |                                                                                  |
| 139 | 0271 | 1 | EXTERNAL                            |                                                                                  |



```

: 140 0272 1 DST$BEGIN_ADDR,      ! Virtual address of where the DST
: 141 0273 1                      ! begins
: 142 0274 1 RST$START_ADDR: REF RST$ENTRY; ! Pointer to first Module RST Entry
: 143 0275 1
: 144 0276 1
: 145 0277 1 LITERAL
: 146 0278 1     DBG$K_ADDRKIND_LITERAL = 0,      ! Indicate the content is a constant
: 147 0279 1     DBG$K_ADDRKIND_ADDR = 1,         ! Indicate the content is a address
: 148 0280 1                                     ! representation
: 149 0281 1     DBG$K_ADDRKIND_DESC = 2,         ! Indicate the content is a descriptor
: 150 0282 1                                     ! address representation
: 151 0283 1     DBG$K_ADDRKIND_REG = 3,          ! Indicate the content is a register
: 152 0284 1     DBG$K_ADDRKIND_ABS = 4,          ! Indicate the content is a absolute
: 153 0285 1                                     ! address value
: 154 0286 1     DBG$K_ADDRKIND_BITOFF = 5,       ! Indicate the address is a bit offsets
: 155 0287 1     DBG$K_ADDRKIND_BLIFLD = 6,
: 156 0288 1     DBG$K_ADDRKIND_COMPLEX = 7,      ! Indicate the address is too complex
: 157 0289 1                                     ! to calculate
: 158 0290 1     DBG$K_ADDRKIND_INVALID = 8;      ! Indicate the address is invalid
: 159 0291 1
: 160 0292 1
: 161 0293 1 LITERAL
: 162 0294 1     SYMBOL_TYPE = 1,                ! SHOW SYMBOL command qualifier /TYPE
: 163 0295 1     SYMBOL_ADDRESS = 2,             ! SHOW SYMBOL command qualifier /ADDRESS
: 164 0296 1     SYMBOL_DIRECT = 3,              ! SHOW SYMBOL command qualifier /DIRECT
: 165 0297 1     SYMBOL_RST = 4,                ! SHOW SYMBOL command qualifier /RST
: 166 0298 1     SYMBOL_DST = 5;                ! SHOW SYMBOL command qualifier /DST
: 167 0299 1
: 168 0300 1
: 169 0301 1 OWN
: 170 0302 1     DBG$_PRINT_FLAG,                ! Used to control print routine
: 171 0303 1                                     ! in SHOW_SYMBOL_TYPE
: 172 0304 1     DBG$_QUALIFIER: BITVECTOR[32]    ! Bitvector for command qualifiers
: 173 0305 1         INITIAL (0),
: 174 0306 1     DBG$_SCOPELIST: REF VECTOR[,LONG]; ! Pointer to the scope search list
: 175 0307 1                                     ! for SHOW SYMBOL command
: 176 0308 1
: 177 0309 1
: 178 0310 1 FIELD WILDCARDS$_FLDS_DEF =          ! A linked list used to contain the
: 179 0311 1                                     ! sub-strings broken up from
: 180 0312 1                                     ! wildcard symbol name with no
: 181 0313 1                                     ! "*" character
: 182 0314 1
: 183 0315 1     SET
: 184 0316 1     WILDCARDS$_LINK = [0, L_],      ! Pointer to next node
: 185 0317 1     WILDCARDS$_STRPTR = [1, L_],    ! Pointer to Counted ASCII Sub-String
: 186 0318 1     TES;
: 187 0319 1 LITERAL
: 188 0320 1     WILDCARDS$_SIZE = 2;             ! Size of link node in longword
: 189 0321 1
: 190 0322 1 MACRO
: 191 0323 1     WILDCARDS$_FLDS = BLOCK[WILDCARDS$_SIZE] FIELD(WILDCARDS$_FLDS_DEF) %;
: 192 0324 1
: 193 0325 1
: 194 0326 1 MACRO
: 195 M 0327 1     INIT_DESCRIPTOR(DESCR) =
: 196 M 0328 1     DESCR[DSC$_LENGTH] = 0;
```



```
197 M 0329 1 DESCR[DSC$B_DTYPE] = DSC$K_DTYPE_T;
198 M 0330 1 DESCR[DSC$B_CLASS] = DSC$K_CLASS_D;
199 M 0331 1 DESCR[DSC$A_POINTER] = 0;%;
200 M 0332 1
201 M 0333 1 DISCARD_DESCRIPTOR(DESCR) =
202 M 0334 1 BEGIN
203 M 0335 1 LOCAL
204 M 0336 1 FREE_STATUS;
205 M 0337 1
206 M 0338 1 FREE_STATUS = STR$FREE1_DX(DESCR);
207 M 0339 1 END%;
208 M 0340 1
209 M 0341 1
210 M 0342 1 MACRO
211 M 0343 1 DTYPE_LENGTH(LENGTH) =
212 M 0344 1 IF .LENGTH NEQ 0
213 M 0345 1 THEN
214 M 0346 1 BEGIN
215 M 0347 1 DBG$PRINT(UPLIT BYTE(%ASCIC ' , size:'), 0);
216 M 0348 1 IF .DTYPE EQL DSC$K_DTYPE_V
217 M 0349 1 THEN
218 M 0350 1 DBG$PRINT(UPLIT BYTE(%ASCIC ' !UL bits'), .LENGTH)
219 M 0351 1 ELSE
220 M 0352 1 IF .DTYPE EQL DSC$K_DTYPE_P
221 M 0353 1 THEN
222 M 0354 1 DBG$PRINT(UPLIT BYTE(%ASCIC ' !UL digits'), .LENGTH)
223 M 0355 1 ELSE
224 M 0356 1 DBG$PRINT(UPLIT BYTE(%ASCIC ' !UL bytes'), .LENGTH);
225 M 0357 1 END;%;
226 M 0358 1
227 M 0359 1 DTYPE_DIGITS(DIGITS) =
228 M 0360 1 IF .VAX_DESCR[DSC$B_DIGITS] NEQ 0
229 M 0361 1 THEN
230 M 0362 1 DBG$PRINT(UPLIT BYTE(%ASCIC ' with !UL digits'), .DIGITS);%;
231 M 0363 1
232 M 0364 1 DTYPE_SCALE(SCALE) =
233 M 0365 1 IF .VAX_DESCR[DSC$B_SCALE] NEQ 0
234 M 0366 1 THEN
235 M 0367 1 DBG$PRINT(UPLIT BYTE(%ASCIC ' scaled !SL'), .SCALE)%;
236 M 0368 1
237 M 0369 1
238 M 0370 1 MACRO
239 M 0371 1 BLIUNIT(UNIT_SIZE) =
240 M 0372 1 SELECTONE .UNIT_SIZE OF
241 M 0373 1 SET
242 M 0374 1 [1]:
243 M 0375 1 DBG$PRINT(UPLIT BYTE(%ASCIC 'byte'), 0);
244 M 0376 1 [2]:
245 M 0377 1 DBG$PRINT(UPLIT BYTE(%ASCIC 'word'), 0);
246 M 0378 1 [4]:
247 M 0379 1 DBG$PRINT(UPLIT BYTE(%ASCIC 'long'), 0);
248 M 0380 1 [OTHERWISE]:
249 M 0381 1 DBG$PRINT(UPLIT BYTE(%ASCIC '!SL'), .UNIT_SIZE);
250 M 0382 1 TES%;
251 M 0383 1
252 M 0384 1 MACRO
253 M 0385 1 ITEM_SIZE(BITSIZE) =
```



```

: 254      M 0386 1
: 255      M 0387 1
: 256      M 0388 1
: 257      M 0389 1
: 258      M 0390 1
: 259      M 0391 1
: 260      M 0392 1
: 261      M 0393 1
: 262      M 0394 1
: 263      M 0395 1
: 264      M 0396 1
: 265      M 0397 1
: 266      M 0398 1
: 267      M 0399 1
: 268      M 0400 1
: 269      M 0401 1
: 270      M 0402 1
: 271      M 0403 1
: 272      M 0404 1
: 273      M 0405 1
: 274      M 0406 1
: 275      M 0407 1
: 276      M 0408 1
: 277      M 0409 1

```

```

IF .BITSIZE NEQ 0
THEN
  BEGIN
    LOCAL
      BITS,
      BYTES;

    BYTES = .BITSIZE/8;
    BITS = .BITSIZE AND 7;
    DBG$PRINT(UPLIT BYTE(%ASCIC ' size: '), 0);
    IF (.BITS NEQ 0) AND (.BITSIZE LEQ 32)
    THEN
      DBG$PRINT(UPLIT BYTE(%ASCIC '!UL bits'), .BITSIZE)
    ELSE
      BEGIN
        IF .BYTES NEQ 0
        THEN
          DBG$PRINT(UPLIT BYTE(%ASCIC '!UL bytes'), .BYTES);
        IF .BITS NEQ 0
        THEN
          DBG$PRINT(UPLIT BYTE(%ASCIC '!UL bits'), .BITS);
        END;
      END;
    END%;

```



```
279 0410 1 GLOBAL ROUTINE DBG$STA_ADDR$SPEC(DSTPTR, ADDRPTR, ADDRKIND): =
280 0411 1
281 0412 1 FUNCTION
282 0413 1     This routine interprets the given DST Value Spec and produces the
283 0414 1     corresponding address representation for that data symbol.
284 0415 1
285 0416 1 INPUTS
286 0417 1     DSTPTR - Pointer to DST record.
287 0418 1
288 0419 1     ADDRPTR - The address of a 2-longword vector to receive the address
289 0420 1     specification of the symbol.
290 0421 1
291 0422 1     ADDRKIND - The address of a longword location to receive the address
292 0423 1     kind.
293 0424 1
294 0425 1 OUTPUTS
295 0426 1     ADDRPTR - A pointer to the desired address representation is returned
296 0427 1     to ADDRPTR.
297 0428 1
298 0429 1     ADDRKIND - The kind of the address representation.
299 0430 1
300 0431 1     Return a pointer to VALSPEC of ADDRESS or DESCRIPTOR to the caller
301 0432 1     to continue the unfinished address interpretation.
302 0433 1
303 0434 1
304 0435 2 BEGIN
305 0436 2
306 0437 2 MAP
307 0438 2     ADDRKIND: REF VECTOR[ LONG],      ! Address kind
308 0439 2     ADDRPTR: REF VECTOR[ 2],        ! Address vector
309 0440 2     DSTPTR: REF DST$RECORD;        ! Pointer to DST record
310 0441 2
311 0442 2 LOCAL
312 0443 2     BLITRLR: REF DST$BLI TRAILER1,    ! Pointer to Bliss DST record trailer
313 0444 2     BLIVALSPEC: BLOCK[ 8, BYTE]      ! Value Spec buffer for Bliss Special
314 0445 2     FIELD(DST$VS_HDR_FIELDS),        ! cases DST record
315 0446 2     VALSPEC: REF DST$VAL_SPEC;        ! Pointer to DST Value Spec
316 0447 2
317 0448 2
318 0449 2 ! Get to the Value Specification. Bliss Special Cases DST is a special
319 0450 2 ! case, so handle it differently.
320 0451 2
321 0452 2 IF .DSTPTR[DST$B_TYPE] EQL DST$K_BLI
322 0453 2 THEN
323 0454 2 BEGIN
324 0455 2     BLIVALSPEC[DST$B_VS_VFLAGS] = .DSTPTR[DST$B_BLI_VFLAGS];
325 0456 2     BLITRLR = DSTPTR[DST$A_BLI_TRLR1] + .DSTPTR[DST$B_BLI_LNG];
326 0457 2     BLIVALSPEC[DST$L_VS_VALUE] = .BLITRLR[DST$L_BLI_VALUE];
327 0458 2     VALSPEC = BLIVALSPEC[DST$B_VS_VFLAGS];
328 0459 2 END
329 0460 2
330 0461 2 ELSE
331 0462 2     VALSPEC = DSTPTR[DST$B_VFLAGS];
332 0463 2
333 0464 2
334 0465 2 ! If the value is given by a trailing Value Spec, we get to that Value
335 0466 2 ! Spec. We loop in case the indirection is repeated.
```



```
336 0467 2
337 0468 2
338 0469 2
339 0470 2
340 0471 2
341 0472 2
342 0473 2
343 0474 2
344 0475 2
345 0476 2
346 0477 2
347 0478 2
348 0479 2
349 0480 2
350 0481 2
351 0482 2
352 0483 2
353 0484 2
354 0485 2
355 0486 2
356 0487 2
357 0488 2
358 0489 2
359 0490 2
360 0491 2
361 0492 2
362 0493 2
363 0494 2
364 0495 2
365 0496 2
366 0497 2
367 0498 2
368 0499 2
369 0500 2
370 0501 2
371 0502 2
372 0503 2
373 0504 2
374 0505 2
375 0506 2
376 0507 2
377 0508 2
378 0509 2
379 0510 2
380 0511 2
381 0512 2
382 0513 2
383 0514 2
384 0515 2
385 0516 2
386 0517 2
387 0518 2
388 0519 2
389 0520 2
390 0521 2
391 0522 2
392 0523 3

!
WHILE .VALSPEC[DST$B_VS_VFLAGS] EQL DST$K_VFLAGS_TVS DO
    VALSPEC = VALSPEC[DST$A_VS_TVS_BASE] + .VALSPEC[DST$L_VS_TVS_OFFSET];

! If the Value Spec gives the offset to a descriptor (in the DST), return
! the address of that descriptor to the caller.
IF .VALSPEC[DST$B_VS_VFLAGS] EQL DST$K_VFLAGS_DSC
THEN
    BEGIN
        ADDRPTR[0] = VALSPEC[DST$A_VS_DSC_BASE] + .VALSPEC[DST$L_VS_DSC_OFFSET];
        ADDRPTR[1] = 0;
        ADDRKIND[0] = DBG$K_ADDRKIND_DESC;
        RETURN .VALSPEC;
    END;

! If this is a Bit Offset Value Spec, return that bit offset as a byte
! address plus bit offset to the caller.
IF .VALSPEC[DST$B_VS_VFLAGS] EQL DST$K_VFLAGS_BITOFFS
THEN
    BEGIN
        ADDRPTR[0] = .VALSPEC[DST$L_VS_VALUE]/8;
        ADDRPTR[1] = .VALSPEC[DST$L_VS_VALUE] AND 7;
        ADDRKIND[0] = DBG$K_ADDRKIND_BITOFF;
        RETURN .VALSPEC;
    END;

! If this is a Value-Spec-Follows value spec, a more complex value spec
! follows the VFLAGS field. Here we handle those kinds of value specs.
IF .VALSPEC[DST$B_VS_VFLAGS] EQL DST$K_VS_FOLLOWS
THEN
    BEGIN
        ! Sort out which particular kind of complex value specification follows.
        !
        CASE .VALSPEC[DST$B_VS_ALLOC] FROM DST$K_VS_ALLOC_STAT TO DST$K_VS_ALLOC_DYN OF
            SET
                ! Handle statically or dynamically allocated objects without Binding
                ! Specs. Here we just indicate this is too complex to compute, for
                ! if we do evaluate the Materialization Spec will cause the side
                ! effects.
                [DST$K_VS_ALLOC_STAT,
                 DST$K_VS_ALLOC_DYN]:
                    BEGIN
                        ADDRPTR[0] = 0;
                        ADDRPTR[1] = 0;
                        ADDRKIND[0] = DBG$K_ADDRKIND_COMPLEX;
                    END;
```

```

393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449

0524
0525
0526
0527
0528
0529
0530
0531
0532
0533
0534
0535
0536
0537
0538
0539
0540
0541
0542
0543
0544
0545
0546
0547
0548
0549
0550
0551
0552
0553
0554
0555
0556
0557
0558
0559
0560
0561
0562
0563
0564
0565
0566
0567
0568
0569
0570
0571
0572
0573
0574
0575
0576
0577
0578
0579
0580

! Any other value in the DST$B_VS_ALLOC field is an error.
[INRANGE, OTRANGE]:
    SIGNAL(DBG$_INV DSTREC);

TES;

! We are done with the complex value spec. Return to the caller.
RETURN .VALSPEC;
END;

! This is an ordinary garden variety Value Spec with a normal VFLAGS field
! and a normal VALUE field. If this is a literal, return the address of the
! literal to the caller.
IF .VALSPEC[DST$V_VS_VALKIND] EQL DST$K_VALKIND_LITERAL
THEN
    BEGIN
        ADDRPTR[0] = VALSPEC[DST$L_VS_VALUE];
        ADDRPTR[1] = 0;
        ADDRKIND[0] = DBG$K_ADDRKIND_LITERAL;
        RETURN .VALSPEC;
    END;

! If this is a register number, return the register number.
IF .VALSPEC[DST$V_VS_VALKIND] EQL DST$K_VALKIND_REG
THEN
    BEGIN
        ADDRPTR[0] = .VALSPEC[DST$L_VS_VALUE];
        ADDRPTR[1] = 0;
        ADDRKIND[0] = DBG$K_ADDRKIND_REG;
        RETURN .VALSPEC;
    END;

! This value spec requires the value to be computed. The resulting value
! is either the address of some object or the address of a descriptor.
! Return to the caller to do some more interpretation about the address of
! this value, used -1 to indicate unfinished status, return its kind:
! address or descriptor address.
ADDRPTR[0] = -1;
ADDRPTR[1] = -1;
IF .VALSPEC[DST$V_VS_VALKIND] EQL DST$K_VALKIND_DESC
THEN
    ADDRKIND[0] = DBG$K_ADDRKIND_DESC
ELSE
    ADDRKIND[0] = DBG$K_ADDRKIND_ADDR;
```



DBGSYMBOL  
V04-000

L 12  
16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

VAX-11 Bliss-32 V4.0-742  
[DEBUG.SRC]DBGSYMBOL.B32;1

Page 10  
(3)

: 450 0581 2 RETURN .VALSPEC;  
: 451 0582 2  
: 452 0583 1 END;

.TITLE DBGSYMBOL  
.IDENT \V04-000\  
.PSECT DBG\$OWN,NOEXE, PIC,2

00000 DBG\$\_PRINT\_FLAG:  
00000000 .BLKB 4  
00004 DBG\$\_QUALIFIER:  
00008 DBG\$\_SCOPELIST:  
.LONG 0  
.BLKB 4

.EXTRN DBG\$GET\_DST\_NAME  
.EXTRN DBG\$GET\_MEMORY, DBG\$GET\_TEMPMEM  
.EXTRN DBG\$HASH\_FIND, DBG\$HASH\_FIND\_SETUP  
.EXTRN DBG\$LANGUAGE, DBG\$MAKE\_INTEGER\_DESC  
.EXTRN DBG\$NEWLINE, DBG\$NPATHDESC\_TO\_CS  
.EXTRN DBG\$POP\_TEMPMEM  
.EXTRN DBG\$PRINT, DBG\$PRINT\_CONTROL  
.EXTRN DBG\$PUSH\_TEMPMEM  
.EXTRN DBG\$REL\_MEMORY, DBG\$RST\_BLDSCOPE\_LIST  
.EXTRN DBG\$RST\_DST\_PTR  
.EXTRN DBG\$RST\_TEMP\_RELEASE  
.EXTRN DBG\$SAVE\_VAL, DBG\$STA\_NUMBERED\_SCOPE  
.EXTRN DBG\$STA\_SETCONTEXT  
.EXTRN DBG\$STA\_SYMKIND  
.EXTRN DBG\$STA\_SYMPATHNAME  
.EXTRN DBG\$STA\_SYMTYPE  
.EXTRN DBG\$STA\_TYP\_AREA  
.EXTRN DBG\$STA\_TYP\_ARRAY  
.EXTRN DBG\$STA\_TYP\_ATOMIC  
.EXTRN DBG\$STA\_TYP\_DESCR  
.EXTRN DBG\$STA\_TYP\_ENUM  
.EXTRN DBG\$STA\_TYP\_FILE  
.EXTRN DBG\$STA\_TYP\_OFFSET  
.EXTRN DBG\$STA\_TYP\_PICT  
.EXTRN DBG\$STA\_TYP\_TYPEDPTR  
.EXTRN DBG\$STA\_TYP\_RECORD  
.EXTRN DBG\$STA\_TYP\_SET  
.EXTRN DBG\$STA\_TYP\_SUBRNG  
.EXTRN DBG\$STA\_TYP\_VARIANT  
.EXTRN DBG\$STA\_TYP\_VARIANT\_COMP  
.EXTRN OTSSCVT\_L\_TZ, STR\$CONCAT  
.EXTRN STR\$DUPL\_CHAR, STR\$FREE1\_DX  
.EXTRN DST\$BEGIN\_ADDR, RST\$START\_ADDR

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

SE 0004 00000  
50 08 C2 00002  
04 AC D0 00005  
01 A0 95 00009

.ENTRY DBG\$STA\_ADDRSPEC, Save R2  
SUBL2 #8, SP  
MOVL DSTPTR, R0  
TSTB 1(R0)

: 0410  
: 0452  
:

|    |           |      |    |      |    |       |        |                           |      |  |
|----|-----------|------|----|------|----|-------|--------|---------------------------|------|--|
|    |           |      |    | 16   | 12 | 0000C | BNEQ   | 1\$                       |      |  |
|    |           | 6E   | 04 | A0   | 90 | 0000E | MOVB   | 4(R0), BLIVALSPEC         | 0455 |  |
|    |           | 51   | 02 | A0   | 9A | 00012 | MOVZBL | 2(R0), R1                 | 0456 |  |
|    |           | 50   | 03 | A041 | 9E | 00016 | MOVAB  | 3(R0)[R1], BLITRLR        |      |  |
|    | 01        | AE   |    | 60   | D0 | 0001B | MOVL   | (BLITPLR), BLIVALSPEC+1   | 0457 |  |
|    |           | 52   |    | 6E   | 9E | 0001F | MOVAB  | BLIVALSPEC, VALSPEC       | 0458 |  |
|    |           |      |    | 04   | 11 | 00022 | BRB    | 2\$                       | 0452 |  |
|    |           | 52   | 02 | A0   | 9E | 00024 | MOVAB  | 2(R0), VALSPEC            | 0462 |  |
|    | FB        | 8F   |    | 62   | 91 | 00028 | CMPB   | (VALSPEC), #251           | 0468 |  |
|    |           |      |    | 0B   | 12 | 0002C | BNEQ   | 3\$                       |      |  |
| 51 |           | 52   | 01 | A2   | C1 | 0002E | ADDL3  | 1(VALSPEC), VALSPEC, R1   | 0469 |  |
|    |           | 52   | 05 | A1   | 9E | 00033 | MOVAB  | 5(R1), VALSPEC            |      |  |
|    |           |      |    | EF   | 11 | 00037 | BRB    | 2\$                       |      |  |
|    | FA        | 8F   |    | 62   | 91 | 00039 | CMPB   | (VALSPEC), #250           | 0475 |  |
|    |           |      |    | 13   | 12 | 0003D | BNEQ   | 4\$                       |      |  |
|    |           | 50   | 08 | AC   | D0 | 0003F | MOVL   | ADDRPTR, R0               | 0478 |  |
| 51 |           | 52   | 01 | A2   | C1 | 00043 | ADDL3  | 1(VALSPEC), VALSPEC, R1   |      |  |
|    |           | 60   | 05 | A1   | 9E | 00048 | MOVAB  | 5(R1), (R0)               |      |  |
|    |           |      | 04 | A0   | D4 | 0004C | CLRL   | 4(R0)                     | 0479 |  |
|    |           |      |    | 0085 | 31 | 0004F | BRW    | 11\$                      | 0480 |  |
|    | FF        | 8F   |    | 62   | 91 | 00052 | CMPB   | (VALSPEC), #255           | 0488 |  |
|    |           |      |    | 16   | 12 | 00056 | BNEQ   | 5\$                       |      |  |
|    |           | 50   | 08 | AC   | D0 | 00058 | MOVL   | ADDRPTR, R0               | 0491 |  |
| 04 | A0        |      |    | 08   | C7 | 0005C | DIVL3  | #8, 1(VALSPEC), (R0)      |      |  |
|    | 01        | A2   |    | 00   | EF | 00061 | EXTZV  | #0, #3, 1(VALSPEC), 4(R0) | 0492 |  |
|    | OC        | BC   |    | 05   | D0 | 00068 | MOVL   | #5, @ADDRKIND             | 0493 |  |
|    |           |      |    | 73   | 11 | 0006C | BRB    | 13\$                      | 0494 |  |
|    | FD        | 8F   |    | 62   | 91 | 0006E | CMPB   | (VALSPEC), #253           | 0501 |  |
|    |           |      |    | 24   | 12 | 00072 | BNEQ   | 8\$                       |      |  |
|    | 01        | 01   | 03 | A2   | 8F | 00074 | CASEB  | 3(VALSPEC), #1, #1        | 0508 |  |
|    |           | 0013 |    | 0013 |    | 00079 | .WORD  | 7\$-6\$,-                 |      |  |
|    |           |      |    |      |    |       |        | 7\$-6\$                   |      |  |
|    |           |      |    | 8F   | DD | 0007D | PUSHL  | #164650                   | 0529 |  |
|    | 00000000G | 00   |    | 01   | FB | 00083 | CALLS  | #1, LIB\$SIGNAL           |      |  |
|    |           |      |    | 55   | 11 | 0008A | BRB    | 13\$                      |      |  |
|    |           | 50   | 08 | AC   | D0 | 0008C | MOVL   | ADDRPTR, R0               | 0520 |  |
|    |           |      |    | 60   | 7C | 00090 | CLRQ   | (R0)                      |      |  |
|    | OC        | BC   |    | 07   | D0 | 00092 | MOVL   | #7, @ADDRKIND             | 0522 |  |
|    |           |      |    | 49   | 11 | 00096 | BRB    | 13\$                      | 0536 |  |
|    |           | 03   |    | 62   | 93 | 00098 | BITB   | (VALSPEC), #3             | 0544 |  |
|    |           |      |    | 10   | 12 | 0009B | BNEQ   | 9\$                       |      |  |
|    |           | 50   | 08 | AC   | D0 | 0009D | MOVL   | ADDRPTR, R0               | 0547 |  |
|    |           | 60   | 01 | A2   | 9E | 000A1 | MOVAB  | 1(R2), (R0)               |      |  |
|    |           |      | 04 | A0   | D4 | 000A5 | CLRL   | 4(R0)                     | 0548 |  |
|    |           |      | OC | BC   | D4 | 000A8 | CLRL   | @ADDRKIND                 | 0549 |  |
|    |           |      |    | 34   | 11 | 000AB | BRB    | 13\$                      | 0550 |  |
| 03 |           | 62   | 02 | 00   | ED | 000AD | CMPZV  | #0, #2, (VALSPEC), #3     | 0556 |  |
|    |           |      |    | 11   | 12 | 000B2 | BNEQ   | 10\$                      |      |  |
|    |           | 50   | 08 | AC   | D0 | 000B4 | MOVL   | ADDRPTR, R0               | 0559 |  |
|    |           | 60   | 01 | A2   | D0 | 000B8 | MOVL   | 1(VALSPEC), (R0)          |      |  |
|    |           |      | 04 | A0   | D4 | 000BC | CLRL   | 4(R0)                     | 0560 |  |
|    | OC        | BC   |    | 03   | D0 | 000BF | MOVL   | #3, @ADDRKIND             | 0561 |  |
|    |           |      |    | 1C   | 11 | 000C3 | BRB    | 13\$                      | 0562 |  |
|    |           | 50   | 08 | AC   | D0 | 000C5 | MOVL   | ADDRPTR, R0               | 0572 |  |
|    |           | 60   |    | 01   | CE | 000C9 | MNEGL  | #1, (R0)                  |      |  |
|    | 04        | A0   |    | 01   | CE | 000CC | MNEGL  | #1, 4(R0)                 | 0573 |  |
| 02 |           | 02   |    | 00   | ED | 000D0 | CMPZV  | #0, #2, (VALSPEC), #2     | 0574 |  |



DBGSYMBOL  
V04-000

N 12  
16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

VAX-11 Bliss-32 V4.0-742  
[DEBUG.SRC]DBGSYMBOL.B32;1

Page 12  
(3)

|    |    |    |       |       |       |      |               |   |      |
|----|----|----|-------|-------|-------|------|---------------|---|------|
| OC | BC | 06 | 12    | 000D5 |       | BNEQ | 12\$          | : |      |
|    |    | 02 | D0    | 000D7 | 11\$: | MOVL | #2, @ADDRKIND | : | 0576 |
|    |    | 04 | 11    | 000DB |       | BRB  | 13\$          | : |      |
| OC | BC | 01 | D0    | 000DD | 12\$: | MOVL | #1, @ADDRKIND | : | 0579 |
|    | 50 | 52 | D0    | 000E1 | 13\$: | MOVL | VALSPEC, R0   | : | 0581 |
|    |    | 04 | 000E4 |       |       | RET  |               | : | 0583 |

; Routine Size: 229 bytes,      Routine Base: DBG\$CODE + 0000



```

454 0584 1 GLOBAL ROUTINE DBG$STA_SHOWSYMBOL(VERB_NODE) =
455 0585 1
456 0586 1 FUNCTION
457 0587 1
458 0588 1     SHOW SYMBOL[/qualifier...] namespec [IN scopespec[,scopespec...]]
459 0589 1
460 0590 1     This routine performs the semantic action associated with the
461 0591 1     SHOW SYMBOL command. First step sets up a scope search list for
462 0592 1     the specified symbol name (wildcard scheme is allowed). If
463 0593 1     scopespec is specified, DBG$RST_BLDSCOPE_LIST is called to construct
464 0594 1     the set scope list, and converts all the numbered scopes to module
465 0595 1     RST and scope RST pointers by calling DBG$STA_NUMBERED_SCOPE, then
466 0596 1     constructs a list of search scopes from set scope search list.
467 0597 1     If scopespec is omitted, all set modules and the GST are set up
468 0598 1     in the scope search list. Second step generates the candidate symbols,
469 0599 1     a "candidate symbol" is a symbol whose name matches namespec,
470 0600 1     but which may not be in any of the scopespec scopes. Third
471 0601 1     step selects candidates in the scope search list. Finally,
472 0602 1     the information to be showed could include the symbol name,
473 0603 1     type information, address information, and (for DEBUG developers)
474 0604 1     the symbol RST and DST entries in hexadecimal.
475 0605 1
476 0606 1 INPUTS
477 0607 1     VERB_NODE - A pointer to the Verb Node in the command execution
478 0608 1               tree which was constructed after parsing the SHOW SYMBOL
479 0609 1               command.
480 0610 1
481 0611 1 OUTPUTS
482 0612 1     Return value is TRUE if any matches were found; false otherwise.
483 0613 1
484 0614 1
485 0615 2 BEGIN
486 0616 2
487 0617 2 MAP
488 0618 2     VERB_NODE: REF DBG$VERB_NODE; : Pointer to the Verb Node in the
489 0619 2                                   command execution tree
490 0620 2
491 0621 2 LOCAL
492 0622 2     ADVERB_NODE: REF DBG$ADVERB_NODE; : Pointer to the Adverb Node in the
493 0623 2                                       command execution tree
494 0624 2     BAD_SCOPE, : Pointer to a counted string
495 0625 2                   representing the translation
496 0626 2                   of the contents of the
497 0627 2                   pathname descriptor
498 0628 2
499 0629 2     CNT1, : Temporary count variable
500 0630 2     CNT2, : Temporary count variable
501 0631 2     DATA_LIST: REF VECTOR[,LONG], : A link list of data symbols
502 0632 2     ERRINDEX, : Indicate the bad scope pathname is
503 0633 2                   found
504 0634 2     IN_SCOPE, : Flag set if candidate symbol is in
505 0635 2                   the given scope
506 0636 2     INVOCNUM, : Invocation number for numeric scope
507 0637 2                   (only needed for parameter passing)
508 0638 2     MODRSTPTR, : Pointer to module RST entry in scope
509 0639 2                   search list
510 0640 2     NAMESPEC: REF VECTOR[,BYTE], : Pointer to a counted string of
```



```
511 0641 2
512 0642 2
513 0643 2
514 0644 2
515 0645 2
516 0646 2
517 0647 2
518 0648 2
519 0649 2
520 0650 2
521 0651 2
522 0652 2
523 0653 2
524 0654 2
525 0655 2
526 0656 2
527 0657 2
528 0658 2
529 0659 2
530 0660 2
531 0661 2
532 0662 2
533 0663 2
534 0664 2
535 0665 2
536 0666 2
537 0667 2
538 0668 2
539 0669 2
540 0670 2
541 0671 2
542 0672 2
543 0673 2
544 0674 2
545 0675 2
546 0676 2
547 0677 2
548 0678 2
549 0679 2
550 0680 2
551 0681 2
552 0682 2
553 0683 2
554 0684 2
555 0685 2
556 0686 2
557 0687 2
558 0688 2
559 0689 2
560 0690 2
561 0691 2
562 0692 2
563 0693 2
564 0694 2
565 0695 2
566 0696 2
567 0697 2

NAME: REF VECTOR[.BYTE],
NCANDS,
NEXTSETMOD: REF RST$ENTRY,
NEXTRSTPTR: REF RST$ENTRY,
NOUN_NODE: REF DBG$NOUN_NODE,
RSTPTR: REF RST$ENTRY,
SCOPE: REF RST$ENTRY,
SCOPEPTR: REF SCOPE$ENTRY,
SCOPE$SPEC: REF VECTOR[.LONG],
STATUS,
SYMSCOPE: REF RST$ENTRY,
SYM_SCOPE$LIST: REF SCOPE$ENTRY,
UTILITY_FLAG,
WILDCARD: REF VECTOR[.BYTE],
WILDCARD_NAME: REF VECTOR[.BYTE],
WILDCARD_PTR: REF WILDCARD$_FLDS,

! Set print controls for DBG$PRINT.
DBG$PRINT_CONTROL(DBG$_K_PRTSET_CONTINUE, 2, DBG$_K_PRTBRK_ON_BLANKS);

! Initialize the scope search list pointer, set scope search list pointer,
and command qualifiers bitvector.
DBG$_SCOPELIST = 0;
SYM_SCOPE$LIST = 0;
DBG$_QUALIFIER = 0;

! Recover the Noun Node for namespace and scopespec, and Adverb Node
for command qualifiers from SHOW SYMBOL command execution tree Verb Node.
NOUN_NODE = .VERB_NODE[DBG$_VERB_OBJECT_PTR];
ADVERB_NODE = .VERB_NODE[DBG$_VERB_ADVERB_PTR];

! Set up command qualifiers from Adverb_node.
IF .ADVERB_NODE NEQ 0
THEN
BEGIN

! There are qualifiers specified on the command.
```

```
namespace field in SHOW SYMBOL
command
Pointer to a counted string
Number of scope entries or later on
used as Number of displayed symbols
Pointer to the next set module after
this one
Pointer to the next RST entry belong
to the same module as this one
Pointer to the Noun Node in the
command execution tree
Pointer to candidate RST entry
Pointer to scope RST entry in set
scope search list
Pointer to the set scope search list
Pointer to the pathname descriptor
vector of scopespec field in
SHOW SYMBOL command
Pointer to symbol scope RST entry
Pointer to the set scope search list
Flag set to indicate some condition
is encountered
Pointer to '*' character in a string
String contains '*', with appended
begin and end characters
Pointer to Sub-Strings
```



```

: 568      0698      3      !
: 569      0699      3      WHILE TRUE DO
: 570      0700      4      BEGIN
: 571      0701      4
: 572      0702      4
: 573      0703      4      ! Check to see if there are more qualifiers.
: 574      0704      4      !
: 575      0705      4      IF .ADVERB_NODE EQL 0 THEN EXITLOOP;
: 576      0706      4
: 577      0707      4
: 578      0708      4      ! Set up command qualifier bitvector accordingly.
: 579      0709      4      !
: 580      0710      4      CASE .ADVERB_NODE[DBG$B_ADVERB_LITERAL] FROM
: 581      0711      4      SYMBOL_TYPE TO SYMBOL_DST OF
: 582      0712      4      SET
: 583      0713      4
: 584      0714      4
: 585      0715      4      ! /TYPE
: 586      0716      4      [SYMBOL_TYPE]:
: 587      0717      4      DBG$_QUALIFIER[SYMBOL_TYPE] = TRUE;
: 588      0718      4
: 589      0719      4
: 590      0720      4      ! /ADDRESS
: 591      0721      4      [SYMBOL_ADDRESS]:
: 592      0722      4      DBG$_QUALIFIER[SYMBOL_ADDRESS] = TRUE;
: 593      0723      4
: 594      0724      4
: 595      0725      4      ! /DIRECT
: 596      0726      4      [SYMBOL_DIRECT]:
: 597      0727      4      DBG$_QUALIFIER[SYMBOL_DIRECT] = TRUE;
: 598      0728      4
: 599      0729      4
: 600      0730      4      ! /RST
: 601      0731      4      [SYMBOL_RST]:
: 602      0732      4      DBG$_QUALIFIER[SYMBOL_RST] = TRUE;
: 603      0733      4
: 604      0734      4
: 605      0735      4      ! /DST
: 606      0736      4      [SYMBOL_DST]:
: 607      0737      4      DBG$_QUALIFIER[SYMBOL_DST] = TRUE;
: 608      0738      4
: 609      0739      4      TES;
: 610      0740      4
: 611      0741      4
: 612      0742      4      ! Get next command qualifier.
: 613      0743      4      !
: 614      0744      4      ADVERB_NODE = .ADVERB_NODE[DBG$L_ADVERB_LINK];
: 615      0745      4
: 616      0746      3      END;
: 617      0747      3      ! End of walking through Adverb Node.
: 618      0748      2      END;
: 619      0749      2      ! End of setting up command qualifiers.
: 620      0750      2
: 621      0751      2      ! Pick up scopespec field in the noun node which specifies the
: 622      0752      2      ! pathname descriptor vector for the set scope search list.
: 623      0753      2      ! If there is a set scope search list specified in the command, call
: 624      0754      2      ! build scope list routine to construct the set scope search list.
```



```
!
SCOPE$SPEC = .NOUN_NODE[DBG$L_NOUN_VALUE2];
IF .SCOPE$SPEC NEQ 0
THEN
    BEGIN

        ! Scopespec is specified in the command. Build the set scope search
        ! list.
        SYM_SCOPE$LIST = DBG$RST_BLDSCOPE_LIST(.SCOPE$SPEC, ERRINDEX,
                                                FALSE, FALSE);

        ! Check for bad scope name entry.
        IF .ERRINDEX NEQ 0
        THEN
            BEGIN

                ! Translate the bad scope to a counted string. Signal an
                ! error.
                DBG$NPATHDESC TO CS(.SCOPE$SPEC[.ERRINDEX], BAD_SCOPE);
                SIGNAL(DBG$_NOSUCHSCOPE, 1, .BAD_SCOPE);
            END;
        END;

        ! End of building set scope search list.

        ! Build a list of scopes for the symbols to be searched. Count the number
        ! of scopes in the scope search list.
        NCANDS = 0;

        ! If we have scopespec specified in the command. Convert all the numbered
        ! scopes to module RST and scope RST pointers. If there is one translation
        ! can not be done (returned module RST pointer is zero), an informational
        ! message is signaled. If there are no translations can be done at all,
        ! release memory block, return to the caller. Enter the good scope to
        ! the scope search list.
        IF .SCOPE$SPEC NEQ 0
        THEN
            BEGIN
                CNT1 = 0;
                CNT2 = 0;

                ! Set SCOPEPTR points to the set scope search list.
                SCOPEPTR = .SYM_SCOPE$LIST;
                WHILE .SCOPEPTR-NEQ 0 DO
                    BEGIN
```

```

: 682 0812 4
: 683 0813 4
: 684 0814 4
: 685 0815 4
: 686 0816 4
: 687 0817 4
: 688 0818 4
: 689 0819 4
: 690 0820 4
: 691 0821 4
: 692 0822 4
: 693 0823 4
: 694 0824 4
: 695 0825 4
: 696 0826 4
: 697 0827 5
: 698 0828 5
: 699 0829 5
: 700 0830 5
: 701 0831 5
: 702 0832 5
: 703 0833 5
: 704 0834 5
: 705 0835 5
: 706 0836 5
: 707 0837 5
: 708 0838 6
: 709 0839 6
: 710 0840 6
: 711 0841 6
: 712 0842 5
: 713 0843 5
: 714 0844 4
: 715 0845 4
: 716 0846 4
: 717 0847 4
: 718 0848 4
: 719 0849 4
: 720 0850 4
: 721 0851 4
: 722 0852 4
: 723 0853 4
: 724 0854 4
: 725 0855 4
: 726 0856 4
: 727 0857 4
: 728 0858 4
: 729 0859 4
: 730 0860 4
: 731 0861 4
: 732 0862 4
: 733 0863 4
: 734 0864 4
: 735 0865 3
: 736 0866 3
: 737 0867 3
: 738 0868 3

```

```

! Assume we have a good scope. Count the number of scopes in
! the set scope search list we have encountered. Get the scope
! RST from set scope search list.
UTILITY_FLAG = TRUE;
CNT1 = .CNT1 + 1;
SCOPE = .SCOPEPTR[SCOPE$L_RSTPTR];

! Check to see if this is a numbered scope. If it is, turn it
! into normal scope.
IF .SCOPEPTR[SCOPE$L_STATE] EQL SCOPE$K_NUMBERED
THEN
  BEGIN
    DBG$STA_NUMBERED_SCOPE(.SCOPEPTR[SCOPE$L_MODPTR], MODRSTPTR,
      SCOPE, INVOCNUM);

    ! Check to see if we can convert this numbered scope. If we
    ! cannot, give an informational message. Count the number of
    ! bad scopes. Set the flag to indicate bad status has encountered.
    IF .MODRSTPTR EQL 0
    THEN
      BEGIN
        SIGNAL(DBG$NONUMSCOPE, 1, .SCOPEPTR[SCOPE$L_MODPTR]);
        CNT2 = .CNT2 + 1;
        UTILITY_FLAG = FALSE;
      END;
    END;
    ! End of converting numbered scope.

! Check to see if this is a global scope. If it is then sets up
! scope to be 'ANONYMOUS' module containing all unclaimed global
! symbols.
IF .SCOPEPTR[SCOPE$L_STATE] EQL SCOPE$K_GLOBAL
THEN
  SCOPE = .RST$START_ADDR;

! Enter the good scope to scope search list.
IF .UTILITY_FLAG THEN BLD_SCOPE_SEARCH_LIST(NCANDS, .SCOPE);

! Get next item on set scope search list.
SCOPEPTR = .SCOPEPTR[SCOPE$L_FLINK];
END;
! End of WHILE walking through set
! scope search list.

```



```

: 739      0869      3      ! Check to see if all translations from numeric scope to module scope
: 740      0870      3      ! failed. If it is, release the set scope search list memroy block,
: 741      0871      3      ! return to the caller.
: 742      0872      3
: 743      0873      3      IF .CNT1 EQL .CNT2
: 744      0874      3      THEN
: 745      0875      3          BEGIN
: 746      0876      3              DBG$REL_MEMORY(.SYM_SCOPE$LIST);
: 747      0877      3              RETURN FALSE;
: 748      0878      3              END;
: 749      0879      3
: 750      0880      3      END
: 751      0881      3      ! End of if there is a scopespec
: 752      0882      3      ! specified.
: 753      0883      3
: 754      0884      3      ! If scopespec is omitted. All set modules' RST entries and GST are
: 755      0885      3      ! set up in the scope search list.
: 756      0886      3
: 757      0887      3      ELSE
: 758      0888      3          BEGIN
: 759      0889      3
: 760      0890      3              ! Make NEXTSETMOD point to the first set module.
: 761      0891      3              !
: 762      0892      3              NEXTSETMOD = .RST$START_ADDR;
: 763      0893      3
: 764      0894      3
: 765      0895      3
: 766      0896      3              ! Find the next set module in the module chain. Enter the module
: 767      0897      3              ! RST on scope search list.
: 768      0898      3
: 769      0899      3              WHILE .NEXTSETMOD NEQ 0 DO
: 770      0900      3                  BEGIN
: 771      0901      3                      4
: 772      0902      3                      4
: 773      0903      3                      ! If the module is set then enter the module RST on scope search
: 774      0904      3                      ! list.
: 775      0905      3                      4
: 776      0906      3                      IF .NEXTSETMOD[RST$V_MODSET]
: 777      0907      3                      THEN
: 778      0908      3                          BLD_SCOPE_SEARCH_LIST(NCANDS, .NEXTSETMOD);
: 779      0909      3                      4
: 780      0910      3                      4
: 781      0911      3                      ! Go on to next module in the chain.
: 782      0912      3                      !
: 783      0913      3                      NEXTSETMOD = .NEXTSETMOD[RST$L_NXTMODPTR];
: 784      0914      3                      ! End of setting up all set modules.
: 785      0915      3                      END;
: 786      0916      3
: 787      0917      3              ! Set up Global Symbol Table.
: 788      0918      3              !
: 789      0919      3              BLD_SCOPE_SEARCH_LIST(NCANDS, .RST$START_ADDR);
: 790      0920      3
: 791      0921      3              END;
: 792      0922      3              ! End of if scopespec is omitted.
: 793      0923      3
: 794      0924      3      ! Check to see if we have successfully build scope search list.
: 795      0925      3      ! If not, really something is wrong. If it is ok, enter the scope count
```

```
796 0926 2 ! to the 1st longword in the scope search list.
797 0927 2
798 0928 2
799 0929 2
800 0930 2
801 0931 2
802 0932 2
803 0933 2
804 0934 2
805 0935 2
806 0936 2
807 0937 2
808 0938 2
809 0939 2
810 0940 2
811 0941 2
812 0942 2
813 0943 2
814 0944 2
815 0945 2
816 0946 2
817 0947 2
818 0948 2
819 0949 2
820 0950 2
821 0951 2
822 0952 2
823 0953 2
824 0954 2
825 0955 2
826 0956 4
827 0957 4
828 0958 4
829 0959 4
830 0960 4
831 0961 4
832 0962 4
833 0963 4
834 0964 4
835 0965 4
836 0966 4
837 0967 4
838 0968 4
839 0969 4
840 0970 4
841 0971 4
842 0972 4
843 0973 4
844 0974 4
845 0975 4
846 0976 4
847 0977 4
848 0978 4
849 0979 4
850 0980 4
851 0981 4
852 0982 5

! IF .DBG$_SCOPELIST EQL 0
! THEN
! BEGIN
! IF .SYM_SCOPE$LIST NEQ 0 THEN DBG$REL MEMORY(.SYM_SCOPE$LIST);
! $DBG_ERROR('DBGSYMBOL\DBG$STA_SHOWSYMBOL, err. 10');
! END
! ELSE
!   DBG$_SCOPELIST[0] = .NCANDS;

! Pick up the namespec field which consists of a single symbol name,
! or a symbol name includes wildcard characters. Set up the flag
! accordingly.
NCANDS = 0;
DATA_LIST = .NOUN_NODE[DBG$L_NOUN_VALUE];
WHILE TRUE DO
  BEGIN
    IF .DATA_LIST EQL 0 THEN EXITLOOP;
    NAMESPEC = .DATA_LIST[1];
    WILDCARD = CH$FIND_CH(.NAMESPEC[0], NAMESPEC[1], %C'*');

    ! This symbol name contains wildcard characters. Set the flag to indicate
    ! we have a wildcard symbol name.
    IF .WILDCARD NEQ 0
    THEN
      BEGIN
        UTILITY_FLAG = TRUE;

        ! Append null character before begin character and after end character
        ! in the string, if the begin and end characters are not '*'. Adjust
        ! the count too.
        WILDCARD_NAME = WILDCARD_NAME_SETUP(.NAMESPEC);

        ! Break the wildcard symbol name into sub-strings. ie, A*B*C breaks
        ! into A, B, C.
        WILDCARD_PTR = WILDCARD_SUB_STRING(.WILDCARD_NAME);

        ! Since we have a wildcard symbol name, we have to go through all set
        ! modules following symbol chain pointer for each module, and all
        ! global symbols, check to see if each RST name matches the wildcard
        ! symbol name pattern, and select those declared in the scope. Start
        ! from first Module RST Entry, find the first set module, set the
        ! symbol chain pointer to this as well.
        NEXTSETMOD = .RST$START_ADDR;
        WHILE .NEXTSETMOD NEQ 0 DO
          BEGIN
```



```

: 853      0983  5      IF .NEXTSETMOD[RST$V.MODSET] THEN EXITLOOP;
: 854      0984  5      NEXTSETMOD = .NEXTSETMOD[RST$L_NXTMODPTR];
: 855      0985  4      END;
: 856      0986  4
: 857      0987  4
: 858      0988  4      ! If we have all the module been canceled, set up the pointer to
: 859      0989  4      ! search global symbol only.
: 860      0990  4
: 861      0991  4      IF .NEXTSETMOD EQL 0
: 862      0992  4      THEN
: 863      0993  4          NEXTRSTPTR = .RST$START_ADDR[RST$L_SYMCHNPTR]
: 864      0994  4      ELSE
: 865      0995  4          NEXTRSTPTR = .NEXTSETMOD;
: 866      0996  4
: 867      0997  4      END
: 868      0998  4          ! End of setting up wildcard symbol name.
: 869      0999  4
: 870      1000  4      ! This is a normal symbol name. Set up the name via RST hash table.
: 871      1001  3      ELSE
: 872      1002  4          BEGIN
: 873      1003  4              UTILITY_FLAG = FALSE;
: 874      1004  4              DBG$HASH_FIND_SETUP(.NAMESPEC);
: 875      1005  3          END;
: 876      1006  3
: 877      1007  3
: 878      1008  3
: 879      1009  3      ! Now that we have a scope search list to search. Go through candidate
: 880      1010  3      ! symbols to see if the candidate is in the given scope. A "candidate
: 881      1011  3      ! symbol" is a symbol whose name matches namespec, but which may not be
: 882      1012  3      ! in any of the scopespec scopes.
: 883      1013  3      WHILE TRUE DO
: 884      1014  4          BEGIN
: 885      1015  4
: 886      1016  4
: 887      1017  4              ! Get next candidate. If the flag is set --> this symbol contains
: 888      1018  4              ! wildcard, call WILDCARD routine to handle it.
: 889      1019  4
: 890      1020  4              IF .UTILITY_FLAG
: 891      1021  4              THEN
: 892      1022  4                  RSTPTR = WILDCARD_CANDIDATES(.WILDCARD_PTR, NEXTSETMOD, NEXTRSTPTR)
: 893      1023  4
: 894      1024  4
: 895      1025  4              ! This symbol is normal symbol --> does not have wildcard, call
: 896      1026  4              ! hash routine to handle it.
: 897      1027  4
: 898      1028  4              ELSE
: 899      1029  4                  RSTPTR = DBG$HASH_FIND(.NAMESPEC);
: 900      1030  4
: 901      1031  4
: 902      1032  4              ! Check to see if we have run through all the candidates. If we
: 903      1033  4              ! do then we are all done.
: 904      1034  4
: 905      1035  4              IF .RSTPTR EQL 0 THEN EXITLOOP;
: 906      1036  4
: 907      1037  4
: 908      1038  4              ! Loop through the scope search list to see if this candidate symbol
: 909      1039  4              ! is in scope. Assume we have not found a scope yet.
```

|   |     |      |   |
|---|-----|------|---|
| : | 910 | 1040 | 4 |
| : | 911 | 1041 | 4 |
| : | 912 | 1042 | 4 |
| : | 913 | 1043 | 5 |
| : | 914 | 1044 | 5 |
| : | 915 | 1045 | 5 |
| : | 916 | 1046 | 5 |
| : | 917 | 1047 | 5 |
| : | 918 | 1048 | 5 |
| : | 919 | 1049 | 5 |
| : | 920 | 1050 | 5 |
| : | 921 | 1051 | 5 |
| : | 922 | 1052 | 5 |
| : | 923 | 1053 | 5 |
| : | 924 | 1054 | 5 |
| : | 925 | 1055 | 5 |
| : | 926 | 1056 | 5 |
| : | 927 | 1057 | 6 |
| : | 928 | 1058 | 6 |
| : | 929 | 1059 | 6 |
| : | 930 | 1060 | 6 |
| : | 931 | 1061 | 6 |
| : | 932 | 1062 | 6 |
| : | 933 | 1063 | 6 |
| : | 934 | 1064 | 6 |
| : | 935 | 1065 | 7 |
| : | 936 | 1066 | 7 |
| : | 937 | 1067 | 7 |
| : | 938 | 1068 | 6 |
| : | 939 | 1069 | 6 |
| : | 940 | 1070 | 6 |
| : | 941 | 1071 | 6 |
| : | 942 | 1072 | 6 |
| : | 943 | 1073 | 6 |
| : | 944 | 1074 | 6 |
| : | 945 | 1075 | 6 |
| : | 946 | 1076 | 6 |
| : | 947 | 1077 | 6 |
| : | 948 | 1078 | 6 |
| : | 949 | 1079 | 6 |
| : | 950 | 1080 | 6 |
| : | 951 | 1081 | 6 |
| : | 952 | 1082 | 6 |
| : | 953 | 1083 | 6 |
| : | 954 | 1084 | 6 |
| : | 955 | 1085 | 6 |
| : | 956 | 1086 | 6 |
| : | 957 | 1087 | 6 |
| : | 958 | 1088 | 7 |
| : | 959 | 1089 | 6 |
| : | 960 | 1090 | 6 |
| : | 961 | 1091 | 6 |
| : | 962 | 1092 | 6 |
| : | 963 | 1093 | 6 |
| : | 964 | 1094 | 6 |
| : | 965 | 1095 | 6 |
| : | 966 | 1096 | 6 |

```
!
IN_SCOPE = FALSE;
INCR I FROM 1 TO .DBG$_SCOPELST[0] DO
  BEGIN
```

```
! Select Candidates. Select candidate by following the upscope
! chain from each candidate RST entry, compare to scope RST pointer
! in the scope search list to see if the candidate symbol is in
! the scope. If /DIRECT is specified, select only those candidate
! symbols which are declared directly in the scope, but not those
! declared nested within the scope. CNT1 is used to keep track
! of the number of upscopes.
```

```
SYMSCOPE = .RSTPTR;
CNT1 = 0;
WHILE TRUE DO
  BEGIN
```

```
! Compare to see if this candadite symbol is in scope. If it
! is, set the flag to indicate the case.
```

```
IF .SYMSCOPE EQL .DBG$_SCOPELST[I]
THEN
  BEGIN
    IN_SCOPE = TRUE;
    EXITLOOP;
  END;
```

```
! Check to see if we are at the module level. If we are,
! exit.
```

```
IF .SYMSCOPE[RST$B_KIND] EQL RST$K_MODULE THEN EXITLOOP;
```

```
! If this is a type component, go up-scope once more to its
! type.
```

```
IF .SYMSCOPE[RST$B_KIND] EQL RST$K_TYPCOMP
THEN
  SYMSCOPE = .SYMSCOPE[RST$L_UPSCOPEPTR];
```

```
! If the /DIRECT qualifier switch is on, we only need to go
! up-scope once. Check to see if we have, if so, exit.
```

```
IF .DBG$_QUALIFIER[SYMBOL_DIRECT] AND (.CNT1 EQL 1)
THEN
  EXITLOOP;
```

```
SYMSCOPE = .SYMSCOPE[RST$L_UPSCOPEPTR];
```

```
! Increment the upscope counter.
```



```

: 967      1097 6      CNT1 = .CNT1 + 1;
: 968      1098 5      END;
: 969      1099 5      ! End of WHILE checking candidate
: 970      1100 5      ! symbol in scope.
: 971      1101 5
: 972      1102 5      ! Check to see if we have selected the candidate, if so, exit.
: 973      1103 5      ! Otherwise, check the next scope in the scope search list.
: 974      1104 5      IF .IN_SCOPE THEN EXITLOOP;
: 975      1105 5
: 976      1106 4      END;
: 977      1107 4      ! End of INCR checking candidate symbol
: 978      1108 4      ! in the scope search list.
: 979      1109 4
: 980      1110 4      ! We have found this symbol is in the scope. Before interface to
: 981      1111 4      ! output routine to show the information about this symbol according
: 982      1112 4      ! to the given qualifier, do a check to make sure this symbol has
: 983      1113 4      ! a name, if this symbol does not have a name, we simply ignore it.
: 984      1114 4      ! And also ignore the TYPE RST entry. Only if /TYPE is specified,
: 985      1115 4      ! then the TYPE RST entry will be taken care of.
: 986      1116 4
: 987      1117 4      IF .IN_SCOPE
: 988      1118 4      THEN
: 989      1119 5      BEGIN
: 990      1120 5      NAME = DBG$GET_DST_NAME(.RSTPTR[RST$L_DSTPTR]);
: 991      1121 5      IF .NAME[0] NEQ 0 AND .RSTPTR[RST$B_KIND] NEQ RST$K_TYPE
: 992      1122 5      THEN
: 993      1123 6      BEGIN
: 994      1124 6      SHOW_SYMBOL(.RSTPTR, NCANDS);
: 995      1125 6      DATA_LIST[2] = .NCANDS;
: 996      1126 5      END;
: 997      1127 4      END;
: 998      1128 4
: 999      1129 3      END;
: 1000     1130 3      ! End of WHILE selecting all candidate
: 1001     1131 3      ! symbols.
: 1002     1132 3
: 1003     1133 3      IF .UTILITY_FLAG THEN DBG$REL_MEMORY(.WILDCARD_NAME);
: 1004     1134 2      DATA_LIST = .DATA_LIST[0];
: 1005     1135 2      END;
: 1006     1136 2      ! End of WHILE data symbol list.
: 1007     1137 2
: 1008     1138 2
: 1009     1139 2      ! All done. Release the memory blocks.
: 1010     1140 2      IF .SYM_SCOPE$LIST NEQ 0 THEN DBG$REL_MEMORY(.SYM_SCOPE$LIST);
: 1011     1141 2      DBG$REL_MEMORY(.DBG$_SCOPE$LIST);
: 1012     1142 2
: 1013     1143 2      ! If we have found any candidates then return TRUE.
: 1014     1144 2
: 1015     1145 2      RETURN .NCANDS GTR 0;
: 1016     1146 1      END;
```

.PSECT DBG\$PLIT, NOWRT, SHR, PIC, 0

```
24 47 42 44 5C 4C 4F 42 4D 59 53 47 42 44 25 00000 P.AAA: .ASCII \%DBGSYMBOL\<92>\DBG$STA_SHOWSYMBOL. err\
2C 4C 4F 42 4D 59 53 57 4F 48 53 5F 41 54 53 0000F
```

1

CLRL NCANDS : 0789

.....D V



|           |    |           |    |       |       |        |                             |      |
|-----------|----|-----------|----|-------|-------|--------|-----------------------------|------|
| 79        |    |           | 54 | E9    | 000AC | BLBC   | R4, 16\$                    | 0799 |
|           |    |           | 57 | D4    | 000AF | CLRL   | CNT1                        | 0802 |
|           |    |           | 54 | D4    | 000B1 | CLRL   | CNT2                        | 0803 |
| 52        |    |           | 59 | D0    | 000B3 | MOVL   | SYM_SCOPE\$LIST, SCOPEPTR   | 0808 |
|           |    |           | 5F | 13    | 000B6 | BEQL   | 15\$                        | 0809 |
| 56        |    |           | 01 | D0    | 000B8 | MOVL   | #1, UTILITY_FLAG            | 0817 |
|           |    |           | 57 | D6    | 000BB | INCL   | CNT1                        | 0818 |
| 14        | AE | 08        | A2 | D0    | 000BD | MOVL   | 8(SCOPEPTR), SCOPE          | 0819 |
|           | 02 | 04        | A2 | D1    | 000C2 | CMPL   | 4(SCOPEPTR), #2             | 0825 |
|           |    |           | 2E | 12    | 000C6 | BNEQ   | 12\$                        |      |
|           |    | 10        | AE | 9F    | 000C8 | PUSHAB | INVOCNUM                    | 0828 |
|           |    | 18        | AE | 9F    | 000CB | PUSHAB | SCOPE                       |      |
|           |    | 20        | AE | 9F    | 000CE | PUSHAB | MODRSTPTR                   |      |
|           |    | 0C        | A2 | DD    | 000D1 | PUSHL  | 12(SCOPEPTR)                |      |
| 00000000G | 00 |           | 04 | FB    | 000D4 | CALLS  | #4, DBG\$STA_NUMBERED_SCOPE |      |
|           |    | 18        | AE | D5    | 000DB | TSTL   | MODRSTPTR                   | 0836 |
|           |    |           | 16 | 12    | 000DE | BNEQ   | 12\$                        |      |
|           |    | 0C        | A2 | DD    | 000E0 | PUSHL  | 12(SCOPEPTR)                | 0839 |
|           |    |           | 01 | DD    | 000E3 | PUSHL  | #1                          |      |
| 00000000G | 00 | 0002867B  | 8F | DD    | 000E5 | PUSHL  | #165499                     |      |
|           |    |           | 03 | FB    | 000EB | CALLS  | #3, LIB\$SIGNAL             |      |
|           |    |           | 54 | D6    | 000F2 | INCL   | CNT2                        | 0840 |
|           |    |           | 56 | D4    | 000F4 | CLRL   | UTILITY_FLAG                | 0841 |
|           | 03 | 04        | A2 | D1    | 000F6 | CMPL   | 4(SCOPEPTR), #3             | 0851 |
|           |    |           | 08 | 12    | 000FA | BNEQ   | 13\$                        |      |
| 14        | AE | 00000000G | 00 | D0    | 000FC | MOVL   | RST\$START_ADDR, SCOPE      | 0853 |
|           | 0B |           | 56 | E9    | 00104 | BLBC   | UTILITY_FLAG, 14\$          | 0858 |
|           |    | 14        | AE | DD    | 00107 | PUSHL  | SCOPE                       |      |
|           |    | 28        | AE | 9F    | 0010A | PUSHAB | NCANDS                      |      |
| 0000V     | CF |           | 02 | FB    | 0010D | CALLS  | #2, BLD_SCOPE_SEARCH_LIST   |      |
|           | 52 |           | 62 | D0    | 00112 | MOVL   | (SCOPEPTR), SCOPEPTR        | 0863 |
|           |    |           | 9F | 11    | 00115 | BRB    | 11\$                        | 0809 |
|           | 54 |           | 57 | D1    | 00117 | CMPL   | CNT1, CNT2                  | 0873 |
|           |    |           | 3D | 12    | 0011A | BNEQ   | 20\$                        |      |
|           |    |           | 59 | DD    | 0011C | PUSHL  | SYM_SCOPE\$LIST             | 0876 |
| 00000000G | 00 |           | 01 | FB    | 0011E | CALLS  | #1, DBG\$REL_MEMORY         |      |
|           |    | 019B      | 31 | 00125 | BRW   | 48\$   |                             | 0877 |
| 20        | AE | 00000000G | 00 | D0    | 00128 | MOVL   | RST\$START_ADDR, NEXTSETMOD | 0893 |
|           | 52 | 20        | AE | D0    | 00130 | MOVL   | NEXTSETMOD, R2              | 0899 |
|           |    |           | 15 | 13    | 00134 | BEQL   | 19\$                        |      |
|           | 0A | 28        | A2 | E9    | 00136 | BLBC   | 40(R2), 18\$                | 0906 |
|           |    |           | 52 | DD    | 0013A | PUSHL  | R2                          | 0908 |
|           |    | 28        | AE | 9F    | 0013C | PUSHAB | NCANDS                      |      |
| 0000V     | CF |           | 02 | FB    | 0013F | CALLS  | #2, BLD_SCOPE_SEARCH_LIST   |      |
| 20        | AE | 10        | A2 | D0    | 00144 | MOVL   | 16(R2), NEXTSETMOD          | 0913 |
|           |    |           | E5 | 11    | 00149 | BRB    | 17\$                        | 0899 |
|           |    | 00000000G | 00 | DD    | 0014B | PUSHL  | RST\$START_ADDR             | 0919 |
|           |    | 28        | AE | 9F    | 00151 | PUSHAB | NCANDS                      |      |
| 0000V     | CF |           | 02 | FB    | 00154 | CALLS  | #2, BLD_SCOPE_SEARCH_LIST   |      |
|           | 50 | 00000000' | EF | D0    | 00159 | MOVL   | DBG\$SCOPE\$LIST, R0        | 0928 |
|           |    |           | 24 | 12    | 00160 | BNEQ   | 22\$                        |      |
|           |    |           | 59 | D5    | 00162 | TSTL   | SYM_SCOPE\$LIST             | 0931 |
|           |    |           | 09 | 13    | 00164 | BEQL   | 21\$                        |      |
| 00000000G | 00 |           | 59 | DD    | 00166 | PUSHL  | SYM_SCOPE\$LIST             |      |
|           |    |           | 01 | FB    | 00168 | CALLS  | #1, DBG\$REL_MEMORY         |      |
|           |    | 00000000' | EF | 9F    | 0016F | PUSHAB | P.AAA                       | 0932 |
|           |    |           | 01 | DD    | 00175 | PUSHL  | #1                          |      |



|    |               |    |           |      |    |       |        |                          |                              |      |
|----|---------------|----|-----------|------|----|-------|--------|--------------------------|------------------------------|------|
|    |               |    | 00028362  | 8F   | DD | 00177 | PUSHL  | #164706                  |                              |      |
|    | 00000000G     | 00 |           | 03   | FB | 0017D | CALLS  | #3, LIB\$SIGNAL          |                              |      |
|    |               | 60 | 24        | 04   | 11 | 00184 | BRB    | 23\$                     |                              | 0928 |
|    |               |    | 24        | AE   | D0 | 00186 | 22\$:  | MOVL                     | NCANDS, (R0)                 | 0935 |
|    |               | 53 |           | AE   | D4 | 0018A | 23\$:  | CLRL                     | NCANDS                       | 0942 |
|    |               |    |           | 63   | D0 | 0018D | 24\$:  | MOVL                     | (NOUN_NODE), DATA_LIST       | 0943 |
|    |               |    |           | 03   | 12 | 00190 |        | BNEQ                     | 25\$                         | 0946 |
|    |               | 55 | 04        | 010A | 31 | 00192 | BRW    | 46\$                     |                              |      |
|    |               |    |           | A3   | D0 | 00195 | 25\$:  | MOVL                     | 4(DATA_LIST), NAMESPEC       | 0947 |
| 01 | A5            | 50 |           | 65   | 9A | 00199 | MOVZBL | (NAMESPEC), R0           |                              | 0948 |
|    |               | 50 |           | 2A   | 3A | 0019C | LOCC   | #42, R0, 1(NAMESPEC)     |                              |      |
|    |               |    |           | 02   | 12 | 001A1 |        | BNEQ                     | 26\$                         |      |
|    |               |    |           | 51   | D4 | 001A3 |        | CLRL                     | R1                           |      |
|    |               | 6E |           | 51   | D0 | 001A5 | 26\$:  | MOVL                     | R1, WILDCARD                 |      |
|    |               |    |           | 43   | 13 | 001A8 |        | BEQL                     | 31\$                         | 0954 |
|    |               | 56 |           | 01   | D0 | 001AA |        | MOVL                     | #1, UTILITY_FLAG             | 0957 |
|    |               |    |           | 55   | DD | 001AD |        | PUSHL                    | NAMESPEC                     | 0964 |
|    | 0000V         | CF |           | 01   | FB | 001AF | CALLS  | #1, WILDCARD_NAME_SETUP  |                              |      |
|    |               | 5A |           | 50   | D0 | 001B4 | MOVL   | R0, WILDCARD_NAME        |                              |      |
|    |               |    |           | 5A   | DD | 001B7 | PUSHL  | WILDCARD_NAME            |                              | 0970 |
|    | 0000V         | CF |           | 01   | FB | 001B9 | CALLS  | #1, WILDCARD_SUB_STRING  |                              |      |
|    |               | 5B |           | 50   | D0 | 001BE | MOVL   | R0, WILDCARD_PTR         |                              |      |
|    |               | 51 | 00000000G | 00   | D0 | 001C1 | MOVL   | RST\$START_ADDR, R1      |                              | 0980 |
|    | 20            | AE |           | 51   | D0 | 001C8 | MOVL   | R1, NEXTSETMOD           |                              |      |
|    |               | 50 | 20        | AE   | D0 | 001CC | 27\$:  | MOVL                     | NEXTSETMOD, R0               | 0981 |
|    |               |    |           | 0D   | 13 | 001D0 |        | BEQL                     | 29\$                         |      |
|    |               | 07 | 28        | A0   | E8 | 001D2 |        | BLBS                     | 40(R0), 28\$                 | 0983 |
|    | 20            | AE | 10        | A0   | D0 | 001D6 |        | MOVL                     | 16(R0), NEXTSETMOD           | 0984 |
|    |               |    |           | EF   | 11 | 001DB |        | BRB                      | 27\$                         | 0981 |
|    |               |    |           | 07   | 12 | 001DD | 28\$:  | BNEQ                     | 30\$                         | 0991 |
|    | 1C            | AE | 08        | A1   | D0 | 001DF | 29\$:  | MOVL                     | 8(R1), NEXTSTPTR             | 0993 |
|    |               |    |           | 12   | 11 | 001E4 |        | BRB                      | 32\$                         |      |
|    | 1C            | AE | 20        | AE   | D0 | 001E6 | 30\$:  | MOVL                     | NEXTSETMOD, NEXTSTPTR        | 0995 |
|    |               |    |           | 0B   | 11 | 001EB |        | BRB                      | 32\$                         | 0954 |
|    |               |    |           | 56   | D4 | 001ED | 31\$:  | CLRL                     | UTILITY_FLAG                 | 1003 |
|    |               |    |           | 55   | DD | 001EF |        | PUSHL                    | NAMESPEC                     | 1004 |
|    | 00000000G     | 00 |           | 01   | FB | 001F1 | CALLS  | #1, DBG\$HASH_FIND_SETUP |                              |      |
|    |               | 0F |           | 56   | E9 | 001F8 | 32\$:  | BLBC                     | UTILITY_FLAG, 33\$           | 1020 |
|    |               |    | 1C        | AE   | 9F | 001FB |        | PUSHAB                   | NEXTSTPTR                    | 1022 |
|    |               |    | 24        | AE   | 9F | 001FE |        | PUSHAB                   | NEXTSETMOD                   |      |
|    |               |    |           | 5B   | DD | 00201 |        | PUSHL                    | WILDCARD_PTR                 |      |
|    | 0000V         | CF |           | 03   | FB | 00203 | CALLS  | #3, WILDCARD_CANDIDATES  |                              |      |
|    |               |    |           | 09   | 11 | 00208 |        | BRB                      | 34\$                         |      |
|    |               |    |           | 55   | DD | 0020A | 33\$:  | PUSHL                    | NAMESPEC                     | 1029 |
|    | 00000000G     | 00 |           | 01   | FB | 0020C | CALLS  | #1, DBG\$HASH_FIND       |                              |      |
|    |               | 54 |           | 50   | D0 | 00213 | 34\$:  | MOVL                     | R0, RSTPTR                   |      |
|    |               |    |           | 78   | 13 | 00216 |        | BEQL                     | 44\$                         | 1035 |
|    |               |    |           | 58   | D4 | 00218 |        | CLRL                     | IN_SCOPE                     | 1041 |
|    |               |    |           | 50   | D4 | 0021A |        | CLRL                     | I                            | 1088 |
|    |               |    |           | 3C   | 11 | 0021C |        | BRB                      | 41\$                         |      |
|    |               | 52 |           | 54   | D0 | 0021E | 35\$:  | MOVL                     | RSTPTR, SYMSCOPE             | 1054 |
|    |               |    |           | 57   | D4 | 00221 |        | CLRL                     | CNT1                         | 1055 |
|    | 00000000'FF40 |    |           | 52   | D1 | 00223 | 36\$:  | CMPL                     | SYMSCOPE, @DBG\$SCOPELIST[1] | 1063 |
|    |               |    |           | 05   | 12 | 0022B |        | BNEQ                     | 37\$                         |      |
|    |               | 58 |           | 01   | D0 | 0022D |        | MOVL                     | #1, IN_SCOPE                 | 1066 |
|    |               |    |           | 25   | 11 | 00230 |        | BRB                      | 40\$                         | 1065 |
|    |               | 01 | 14        | A2   | 91 | 00232 | 37\$:  | CMPL                     | 20(SYMSCOPE), #1             | 1074 |



|              |              |           |                |        |                           |      |
|--------------|--------------|-----------|----------------|--------|---------------------------|------|
|              |              |           | 1F 13 00236    | BEQL   | 40\$                      |      |
|              | 0A           | 14        | A2 91 00238    | CMPB   | 20(SYMSCOPE), #10         | 1080 |
|              |              |           | 04 12 0023C    | BNEQ   | 38\$                      |      |
|              | 52           | 10        | A2 D0 0023E    | MOVL   | 16(SYMSCOPE), SYMSCOPE    | 1082 |
| 05 00000000' | EF           |           | 03 E1 00242    | BBC    | #3, DBG\$-QUALIFIER, 39\$ | 1088 |
|              | 01           |           | 57 D1 0024A    | CMPL   | CNT1, #1                  |      |
|              |              |           | 08 13 0024D    | BEQL   | 40\$                      |      |
|              | 52           | 10        | A2 D0 0024F    | MOVL   | 16(SYMSCOPE), SYMSCOPE    | 1092 |
|              |              |           | 57 D6 00253    | INCL   | CNT1                      | 1097 |
|              |              |           | CC 11 00255    | BRB    | 36\$                      | 1056 |
|              | 0B           |           | 58 E8 00257    | BLBS   | IN SCOPE, 42\$            | 1104 |
| BC           | 50 00000000' |           | FF F3 0025A    | AOBLEQ | @DBG\$ SCOPELIST, 1, 35\$ | 1042 |
|              | 93           |           | 58 E9 00262    | BLBC   | IN SCOPE, 32\$            | 1117 |
|              |              | 0C        | A4 DD 00265    | PUSHL  | 12(RSTPTR)                | 1120 |
| 00000000G    | 00           |           | 01 FB 00268    | CALLS  | #1, DBG\$GET_DST_NAME     |      |
| 04           | AE           |           | 50 D0 0026F    | MOVL   | R0, NAME                  |      |
|              |              | 04        | BE 95 00273    | TSTB   | @NAME                     | 1121 |
|              |              |           | 80 13 00276    | BEQL   | 32\$                      |      |
|              | 07           | 14        | A4 91 00278    | CMPB   | 20(RSTPTR), #7            |      |
|              |              |           | 0F 13 0027C    | BEQL   | 43\$                      |      |
|              |              | 24        | AE 9F 0027E    | PUSHAB | NCANDS                    | 1124 |
|              |              |           | 54 DD 00281    | PUSHL  | RSTPTR                    |      |
| 0000V        | CF           |           | 02 FB 00283    | CALLS  | #2, SHOW SYMBOL           |      |
| 08           | A3           | 24        | AE D0 00288    | MOVL   | NCANDS, 8(DATA_LIST)      | 1125 |
|              |              |           | FF 68 31 0028D | BRW    | 32\$                      | 1013 |
|              | 09           |           | 56 E9 00290    | BLBC   | UTILITY FLAG, 45\$        | 1132 |
|              |              |           | 5A DD 00293    | PUSHL  | WILDCARD NAME             |      |
| 00000000G    | 00           |           | 01 FB 00295    | CALLS  | #1, DBG\$REL_MEMORY       |      |
|              |              |           | FE EE 31 0029C | BRW    | 24\$                      | 1133 |
|              |              |           | 59 D5 0029F    | TSTL   | SYM_SCOPE\$LIST           | 1139 |
|              |              |           | 09 13 002A1    | BEQL   | 47\$                      |      |
|              |              |           | 59 DD 002A3    | PUSHL  | SYM_SCOPE\$LIST           |      |
| 00000000G    | 00           |           | 01 FB 002A5    | CALLS  | #1, DBG\$REL_MEMORY       |      |
|              |              | 00000000' | EF DD 002AC    | PUSHL  | DBG\$ SCOPELIST           | 1140 |
| 00000000G    | 00           |           | 01 FB 002B2    | CALLS  | #1, DBG\$REL_MEMORY       |      |
|              |              |           | 50 D4 002B9    | CLRL   | R0                        | 1145 |
|              |              | 24        | AE D5 002BB    | TSTL   | NCANDS                    |      |
|              |              |           | 05 15 002BE    | BLEQ   | 49\$                      |      |
|              |              |           | 50 D6 002C0    | INCL   | R0                        |      |
|              |              |           | 04 002C2       | RET    |                           |      |
|              |              |           | 50 D4 002C3    | CLRL   | R0                        | 1146 |
|              |              |           | 04 002C5       | RET    |                           |      |

; Routine Size: 710 bytes, Routine Base: DBG\$CODE + 00E5



```
1018 1147 1 GLOBAL ROUTINE DBG$STA_SYMADDR(SYMID, ADDRPTR, ADDRKIND): =
1019 1148 1
1020 1149 1 FUNCTION
1021 1150 1     This routine takes a SYMID and gets the address information associated
1022 1151 1     with the input SYMID.  NOTE: This piece of code is taken from
1023 1152 1     DBG$STA_SYMVALUE with some modifications, so any changes made in there
1024 1153 1     that impact the logic should be made in here as well.
1025 1154 1
1026 1155 1 INPUTS
1027 1156 1     SYMID      - Symid ID for the symbol.
1028 1157 1
1029 1158 1     ADDRPTR    - The address of a two-longword vector to receive the address
1030 1159 1                 specification of the symbol.
1031 1160 1
1032 1161 1     ADDRKIND   - The address of a longword location to receive the address
1033 1162 1                 kind.
1034 1163 1
1035 1164 1 OUTPUTS
1036 1165 1     ADDRPTR    - A pointer to the desired address is returned to ADDRPTR.
1037 1166 1
1038 1167 1     ADDRKIND   - The kind of the address is returned to ADDRKIND.
1039 1168 1
1040 1169 1     Return 0 or a pointer to VALSPEC for a data symbol to the caller to
1041 1170 1     continue unfinished address interpretation.
1042 1171 1
1043 1172 1 BEGIN
1044 1173 2
1045 1174 2 MAP
1046 1175 2     ADDRKIND: REF VECTOR[1],      ! Address kind
1047 1176 2     ADDRPTR:  REF VECTOR[2],      ! Address vector
1048 1177 2     SYMID:  REF RST$ENTRY;       ! Pointer to symbol's RST entry
1049 1178 2
1050 1179 2 LOCAL
1051 1180 2     DSTPTR: REF DST$RECORD,       ! Pointer to Symbol's DST record
1052 1181 2     PSECT TRAILER: REF DST$PSECT_TRAILER,
1053 1182 2     STKFMPTR: REF VECTOR[.LONG], ! Stack frame pointer from Stack Machine
1054 1183 2                                     (needed for passing parameters)
1055 1184 2     VALPTR: REF DST$VAL_SPEC;     ! Pointer to VALSPEC
1056 1185 2
1057 1186 2
1058 1187 2 VALPTR = 0;
1059 1188 2 CASE .SYMID[RST$B_KIND] FROM RST$K_KIND_MINIMUM TO RST$K_KIND_MAXIMUM OF
1060 1189 2 SET
1061 1190 2
1062 1191 2
1063 1192 2     ! For instruction addresses, return the start address in the RST entry
1064 1193 2     ! in ADDRPTR[0].  If it is a lexical entity, return the end address in
1065 1194 2     ! RST entry in ADDRPTR[1].
1066 1195 2
1067 1196 2     [RST$K_ROUTINE, RST$K_BLOCK,
1068 1197 2     RST$K_LINE]:
1069 1198 2     BEGIN
1070 1199 2         ADDRPTR[0] = .SYMID[RST$L_STARTADDR];
1071 1200 2         ADDRPTR[1] = .SYMID[RST$L_ENDADDR];
1072 1201 2         ADDRKIND[0] = DBG$K_ADDRKIND_ABS;
1073 1202 2     END;
1074 1203 2
```



```
1075 1204 2
1076 1205 2
1077 1206 2
1078 1207 2
1079 1208 2
1080 1209 2
1081 1210 2
1082 1211 2
1083 1212 2
1084 1213 2
1085 1214 2
1086 1215 2
1087 1216 2
1088 1217 2
1089 1218 2
1090 1219 2
1091 1220 2
1092 1221 2
1093 1222 2
1094 1223 2
1095 1224 2
1096 1225 2
1097 1226 2
1098 1227 2
1099 1228 2
1100 1229 2
1101 1230 2
1102 1231 2
1103 1232 2
1104 1233 2
1105 1234 2
1106 1235 2
1107 1236 2
1108 1237 2
1109 1238 2
1110 1239 2
1111 1240 2
1112 1241 2
1113 1242 2
1114 1243 2
1115 1244 2
1116 1245 2
1117 1246 2
1118 1247 2
1119 1248 2
1120 1249 2
1121 1250 2
1122 1251 2
1123 1252 2
1124 1253 2
1125 1254 2
1126 1255 2
1127 1256 2
1128 1257 2
1129 1258 2
1130 1259 2
1131 1260 2
```

```
! For instruction addresses, return the start address in the RST entry
! in ADDRPTR[0], and 0 in ADDRPTR[1].
[RST$K_ENTRY, RST$K_LABEL]:
  BEGIN

  ! Handle the PSECT DST record. Here we pick the PSECT start
  ! address directly from the DST record.
  DSTPTR = .SYMID[RST$L_DSTPTR];
  IF .DSTPTR[DST$B_TYPE] EQL DST$K_PSECT
  THEN
    BEGIN
      PSECT_TRAILER = DSTPTR[DST$A_PSECT_TRLR_BASE] +
        .DSTPTR[DST$B_PSECT_TRLR_OFFS];
      ADDRPTR[0] = .DSTPTR[DST$L_PSECT_VALUE];

      ! Get to the PSECT trailing record to pick up the size.
      ADDRPTR[1] = .ADDRPTR[0] + .PSECT_TRAILER[DST$L_PSECT_SIZE] - 1;
    END
  ELSE
    BEGIN
      ADDRPTR[0] = .SYMID[RST$L_STARTADDR];
      ADDRPTR[1] = 0;
    END;

  ADDRKIND[0] = DBG$K_ADDRKIND_ABS;
  END;

! For Data and Type Components, we determine the address from DST
! value specification.
[RST$K_DATA, RST$K_TYPCOMP]:
  BEGIN
    DSTPTR = .SYMID[RST$L_DSTPTR];
    CASE .DSTPTR[DST$B_TYPE] FROM 0 TO 255 OF
      SET

      ! Handle all normal DST records, i.e., those of the standard
      ! format. Find the Value Spec and pass it to DBG$STA_VALADDR
      ! for address computation.
      [DSC$K_DTYPE_LOWEST TO DSC$K_DTYPE_HIGHEST,
       DST$K_BOOL, DST$K_SEPTYP, DST$K_LBLORLIT,
       DST$K_ENTRY, DST$K_RTNBEG, DST$K_BLKBEGB,
       DST$K_RECBEGB, DST$K_ENUMELT]:
        BEGIN
          VALPTR = DBG$STA_ADDRSPEC(.DSTPTR, .ADDRPTR, .ADDRKIND);
        END;
```



```
: 1132 1261 3
: 1133 1262 3
: 1134 1263 3
: 1135 1264 3
: 1136 1265 3
: 1137 1266 3
: 1138 1267 3
: 1139 1268 4
: 1140 1269 4
: 1141 1270 4
: 1142 1271 4
: 1143 1272 3
: 1144 1273 3
: 1145 1274 3
: 1146 1275 3
: 1147 1276 3
: 1148 1277 3
: 1149 1278 4
: 1150 1279 4
: 1151 1280 3
: 1152 1281 3
: 1153 1282 3
: 1154 1283 3
: 1155 1284 3
: 1156 1285 3
: 1157 1286 4
: 1158 1287 4
: 1159 1288 4
: 1160 1289 4
: 1161 1290 3
: 1162 1291 3
: 1163 1292 3
: 1164 1293 3
: 1165 1294 3
: 1166 1295 3
: 1167 1296 3
: 1168 1297 4
: 1169 1298 4
: 1170 1299 4
: 1171 1300 4
: 1172 1301 3
: 1173 1302 3
: 1174 1303 3
: 1175 1304 3
: 1176 1305 3
: 1177 1306 3
: 1178 1307 4
: 1179 1308 4
: 1180 1309 4
: 1181 1310 4
: 1182 1311 3
: 1183 1312 3
: 1184 1313 3
: 1185 1314 3
: 1186 1315 2
: 1187 1316 2
: 1188 1317 2
```

```
! Handle the Label DST record. Here we get the label address
! directly from the DST$$_VALUE field -- the DST$$_VFLAGS is
! not provided.
```

```
[DST$$_LABEL]:
  BEGIN
    ADDRPTR[0] = .DSTPTR[DST$$_VALUE];
    ADDRPTR[1] = 0;
    ADDRKIND[0] = DBG$$_ADDRKIND_ABS;
  END;
```

```
! Handle the Bliss special Cases DST record.
```

```
[DST$$_BLI]:
  BEGIN
    VALPTR = DBG$$_STA_ADDRSPEC(.DSTPTR, .ADDRPTR, .ADDRKIND);
  END;
```

```
! Handle the Bliss Field DST record.
```

```
[DST$$_BLIFLD]:
  BEGIN
    ADDRPTR[0] = 0;
    ADDRPTR[1] = 0;
    ADDRKIND[0] = DBG$$_ADDRKIND_BLIFLD;
  END;
```

```
! Handle the COBOL Hack DST Record. Here we just indicate
! this is too complex to compute.
```

```
[DST$$_COB_HACK]:
  BEGIN
    ADDRPTR[0] = 0;
    ADDRPTR[1] = 0;
    ADDRKIND[0] = DBG$$_ADDRKIND_COMPLEX;
  END;
```

```
! Any other DST record treated as invalid address.
```

```
[INRANGE, OUTFRANGE]:
  BEGIN
    ADDRPTR[0] = 0;
    ADDRPTR[1] = 0;
    ADDRKIND[0] = DBG$$_ADDRKIND_INVALID;
  END;
```

```
TES;
```

```
END;
```



```

! Any other RST entry treated as invalid address.
!
[INRANGE, OUTRANGE]:
    BEGIN
        ADDRPTR[0] = 0;
        ADDRPTR[1] = 0;
        ADDRKIND[0] = DBG$K_ADDRKIND_INVALID;
    END;

TES;

RETURN .VALPTR;

END;
```

|      |      |      |    |      |       |        |                                 |        |
|------|------|------|----|------|-------|--------|---------------------------------|--------|
|      |      |      |    | 001C | 00000 | .ENTRY | DBG\$STA_SYMADDR, Save R2,R3,R4 | : 1147 |
|      |      |      |    | D4   | 00002 | CLRL   | VALPTR                          | : 1188 |
|      |      | 53   | 08 | AC   | 7D    | MOVQ   | ADDRPTR, R3                     | : 1322 |
|      |      | 51   | 04 | AC   | D0    | MOVL   | SYMID, R1                       | : 1189 |
|      |      | 00   | 14 | A1   | 8F    | CASEB  | 20(R1), #0, #13                 | :      |
| 001F | 0D   | 025A |    | 025A | 00011 | .WORD  | 8%-1%, -                        | :      |
| 025A | 001F | 001F |    | 0025 | 00019 |        | 8%-1%, -                        | :      |
| 025A | 0050 | 025A |    | 0025 | 00021 |        | 2%-1%, -                        | :      |
|      |      | 025A |    | 025A | 00029 |        | 2%-1%, -                        | :      |
|      |      |      |    |      |       |        | 3%-1%, -                        | :      |
|      |      |      |    |      |       |        | 2%-1%, -                        | :      |
|      |      |      |    |      |       |        | 6%-1%, -                        | :      |
|      |      |      |    |      |       |        | 8%-1%, -                        | :      |
|      |      |      |    |      |       |        | 3%-1%, -                        | :      |
|      |      |      |    |      |       |        | 8%-1%, -                        | :      |
|      |      |      |    |      |       |        | 6%-1%, -                        | :      |
|      |      |      |    |      |       |        | 8%-1%, -                        | :      |
|      |      |      |    |      |       |        | 8%-1%, -                        | :      |
|      |      |      |    |      |       |        | 8%-1%                           | :      |
|      |      |      |    | 023B | 31    | BRW    | 8%                              | : 1322 |
|      |      | 63   | 18 | A1   | 7D    | MOVQ   | 24(R1), (R3)                    | : 1200 |
|      |      |      |    | 21   | 11    | BRB    | 4%                              | : 1202 |
|      |      | 52   | 0C | A1   | D0    | MOVL   | 12(R1), DSTPTR                  | : 1216 |
| B8   |      | 8F   | 01 | A2   | 91    | CMPB   | 1(DSTPTR), #184                 | : 1217 |
|      |      |      |    | 19   | 12    | BNEQ   | 5%                              | :      |
|      |      | 51   | 07 | A2   | 9A    | MOVZBL | 7(DSTPTR), R1                   | : 1221 |
|      |      | 51   | 08 | A142 | 9E    | MOVAB  | 8(R1)[DSTPTR], PSECT_TRAILER    | : 1220 |
|      |      | 63   | 03 | A2   | D0    | MOVL   | 3(DSTPTR), (R3)                 | : 1222 |
|      | 51   | 63   |    | 61   | C1    | ADDL3  | (PSECT_TRAILER), (R3), R1       | : 1227 |
|      |      | 04   | A3 | FF   | A1    | MOVAB  | -1(R1), 4(R3)                   | :      |
|      |      |      |    | 021E | 31    | BRW    | 11%                             | : 1217 |
|      |      | 63   | 18 | A1   | D0    | MOVL   | 24(R1), (R3)                    | : 1232 |
|      |      |      |    | 0214 | 31    | BRW    | 10%                             | : 1233 |
|      |      |      |    | A1   | D0    | MOVL   | 12(R1), DSTPTR                  | : 1245 |
|      |      | 52   | 0C | A2   | 8F    | CASEB  | 1(DSTPTR), #0, #255             | : 1246 |
|      |      | 00   | 01 |      |       | .WORD  | 12%-7%, -                       | :      |
| 0211 | FF   | 0211 |    | 0211 | 0006B |        | 12%-7%, -                       | :      |
| 0211 | 8F   | 0211 |    | 0211 | 00073 |        | 12%-7%, -                       | :      |
| 0211 |      | 0211 |    | 0211 | 0007B |        | 12%-7%, -                       | :      |



|      |      |      |      |       |            |
|------|------|------|------|-------|------------|
| 0211 | 0211 | 0211 | 0211 | 00083 | 12\$-7\$,- |
| 0211 | 0211 | 0211 | 0211 | 0008B | 12\$-7\$,- |
| 0211 | 0211 | 0211 | 0211 | 00093 | 12\$-7\$,- |
| 0211 | 0211 | 0211 | 0211 | 0009B | 12\$-7\$,- |
| 0211 | 0211 | 0211 | 0211 | 000A3 | 12\$-7\$,- |
| 0211 | 0211 | 0211 | 0211 | 000AB | 12\$-7\$,- |
| 0200 | 0200 | 0211 | 0211 | 000B3 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000BB | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000C3 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000CB | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000D3 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000DB | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000E3 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000EB | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000F3 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 000FB | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00103 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0010B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00113 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0011B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00123 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0012B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00133 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0013B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00143 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0014B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00153 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0015B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00163 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0016B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00173 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0017B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00183 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0018B | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 00193 | 12\$-7\$,- |
| 0200 | 0200 | 0200 | 0200 | 0019B | 8\$-7\$,-  |
| 0200 | 0211 | 0200 | 0200 | 001A3 | 8\$-7\$,-  |
| 0211 | 0200 | 0200 | 0200 | 001AB | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0211 | 001B3 | 8\$-7\$,-  |
| 0211 | 0200 | 0200 | 0200 | 001BB | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 001C3 | 8\$-7\$,-  |
| 0200 | 021F | 0200 | 0211 | 001CB | 8\$-7\$,-  |
| 0219 | 0200 | 0211 | 0200 | 001D3 | 8\$-7\$,-  |
| 0206 | 0211 | 0200 | 0200 | 001DB | 8\$-7\$,-  |
| 0200 | 0211 | 0200 | 0200 | 001E3 | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 001EB | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 001F3 | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 001FB | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 00203 | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 0020B | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 00213 | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 0021B | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 00223 | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 0022B | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 00233 | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 0023B | 8\$-7\$,-  |
| 0200 | 0200 | 0200 | 0200 | 00243 | 8\$-7\$,-  |

.....



Page 32  
(5)



DBGSYMBOL  
V04-000

16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

```
VAX-11 Bliss-32 V4.0-742
[DEBUG.SRC]DBGSYMBOL.B32;1
```

Page 33  
(5)

[illegible]

.....



DBGSYMBOL  
V04-000

J 14  
16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

VAX-11 Bliss-32 v4.0-742  
[DEBUG.SRC]DBGSYMBOL.B32:1

Page 34  
(5)[illegible]

.....



K 14  
16-Sep-1984 02:39:38 VAX-11 Bliss-32 V4.0-742  
14-Sep-1984 12:17:52 [DEBUG.SRC]DBGSYMBOL.B32;1

Page 35  
(5)

; Routine Size: 656 bytes, Routine Base: DBG\$CODE + 03AB

```

1204 1332 1 GLOBAL ROUTINE DBG$TRANSLATE_DESCR(VAX_DESCR): NOVALUE =
1205 1333 1
1206 1334 1 FUNCTION
1207 1335 1     Given pointer to VAX Standard Descriptor, translate its information
1208 1336 1     into readable format.
1209 1337 1
1210 1338 1 INPUT
1211 1339 1     VAX_DESCR - Pointer to VAX Standard Descriptor.
1212 1340 1
1213 1341 1 OUTPUT
1214 1342 1     None.
1215 1343 1
1216 1344 1
1217 1345 2 BEGIN
1218 1346 2
1219 1347 2 MAP
1220 1348 2     VAX_DESCR: REF BLOCK[,BYTE];
1221 1349 2
1222 1350 2 LOCAL
1223 1351 2     BOUNDS_PTR: REF VECTOR,      ! Pointer to Bounds for Array Descriptor
1224 1352 2     DTYPE,                      ! Atomic data types
1225 1353 2     LENGTH;                    ! Length of the object
1226 1354 2
1227 1355 2
1228 1356 2 DTYPE = .VAX_DESCR[DSC$B_DTYPE];
1229 1357 2 LENGTH = .VAX_DESCR[DSC$B_LENGTH];
1230 1358 2 CASE .VAX_DESCR[DSC$B_CLASS] FROM DSC$K_CLASS_S TO DSC$K_CLASS_UA OF
1231 1359 2     SET
1232 1360 2
1233 1361 2
1234 1362 2     ! Scalar, String Descriptor, is used for scalar data and fixed-length
1235 1363 2     ! strings. Any VAX data type can be used with this descriptor, except
1236 1364 2     ! bit unaligned. (DTYPE 34).
1237 1365 2
1238 1366 2 [DSC$K_CLASS_S]:
1239 1367 2     BEGIN
1240 1368 2         IF ((.DTYPE GEQ DSC$K_DTYPE_T) AND (.DTYPE LEQ DSC$K_DTYPE_P)) OR
1241 1369 2             (.DTYPE EQL DSC$K_DTYPE_V) OR
1242 1370 2             (.DTYPE EQL DSC$K_DTYPE_VU)
1243 1371 2         THEN
1244 1372 2             DBG$PRINT(UPBIT BYTE(%ASCIC'string descriptor type,'), 0)
1245 1373 2         ELSE
1246 1374 2             DBG$PRINT(UPBIT BYTE(%ASCIC'scalar descriptor type,'), 0);
1247 1375 2
1248 1376 2         DBG$TRANSLATE_TYPECODE(.DTYPE);
1249 1377 2         DTYPE_LENGTH(LENGTH);
1250 1378 2     END;
1251 1379 2
1252 1380 2
1253 1381 2     ! Dynamic String Descriptor, is used for dynamically allocated strings.
1254 1382 2
1255 1383 2 [DSC$K_CLASS_D]:
1256 1384 2     BEGIN
1257 1385 2         DBG$PRINT(UPBIT BYTE(%ASCIC'dynamic string descriptor type,'), 0);
1258 1386 2         DBG$TRANSLATE_TYPECODE(.DTYPE);
1259 1387 2         DTYPE_LENGTH(LENGTH);
1260 1388 2     END;

```



```
1261 1389
1262 1390
1263 1391
1264 1392
1265 1393
1266 1394
1267 1395
1268 1396
1269 1397
1270 1398
1271 1399
1272 1400
1273 1401
1274 1402
1275 1403
1276 1404
1277 1405
1278 1406
1279 1407
1280 1408
1281 1409
1282 1410
1283 1411
1284 1412
1285 1413
1286 1414
1287 1415
1288 1416
1289 1417
1290 1418
1291 1419
1292 1420
1293 1421
1294 1422
1295 1423
1296 1424
1297 1425
1298 1426
1299 1427
1300 1428
1301 1429
1302 1430
1303 1431
1304 1432
1305 1433
1306 1434
1307 1435
1308 1436
1309 1437
1310 1438
1311 1439
1312 1440
1313 1441
1314 1442
1315 1443
1316 1444
1317 1445

! Array Descriptor.
[DSC$K_CLASS_A, DSC$K_CLASS_NCA, DSC$K_CLASS_UBA]:
  BEGIN
  SELECTONE .VAX_DESCR[DSC$B_CLASS] OF
  SET

    ! Array Descriptor, is used to describe contiguous arrays of atomic
    ! data type or contiguous arrays of fixed length strings.
    [DSC$K_CLASS_A]:
      DBG$PRINT(UPLIT BYTE(%ASCIC'array descriptor type,'), 0);

    ! Noncontiguous Array Descriptor describes an array where the storage
    ! of the array elements may be allocated with a fixed, nonzero number
    ! of bytes separating logically adjacent elements.
    [DSC$K_CLASS_NCA]:
      DBG$PRINT(UPLIT BYTE(%ASCIC 'noncontiguous array descriptor type,'), 0);

    ! Unaligned Bit Array Descriptor, is used to specified an array of
    ! unaligned bit strings.
    [DSC$K_CLASS_UBA]:
      DBG$PRINT(UPLIT BYTE(%ASCIC 'unaligned bit array descriptor type,'), 0);
  TES;

  DBG$PRINT(UPLIT BYTE(%ASCIC ' !UL dimensions,'), .VAX_DESCR[DSC$B_DIMCT]);

  ! Point to the lower and upper bounds of the array.
  !
  DBG$PRINT(UPLIT BYTE(%ASCIC ' bounds: ['), 0);
  BOUNDS_PTR = VAX_DESCR[DSC$B_M1];
  BOUNDS_PTR = BOUNDS_PTR[.VAX_DESCR[DSC$B_DIMCT]];
  INCR BOUND_NO FROM 1 TO .VAX_DESCR[DSC$B_DIMCT] DO
  BEGIN
  IF (.BOUND_NO EQL .VAX_DESCR[DSC$B_DIMCT])
  THEN
    DBG$PRINT(UPLIT BYTE(%ASCIC '!SL:!SL)'),
      .BOUNDS_PTR[((.BOUND_NO - 1) * 2)],
      .BOUNDS_PTR[((.BOUND_NO - 1) * 2) + 1])
  ELSE
    DBG$PRINT(UPLIT BYTE(%ASCIC '!SL:!SL,'),
      .BOUNDS_PTR[((.BOUND_NO - 1) * 2)],
      .BOUNDS_PTR[((.BOUND_NO - 1) * 2) + 1]);
  END;
END;

! Procedure Descriptor, is used to specifies its entry address and
```



```
1318 1446 2 ! function value data type, if any.
1319 1447
1320 1448 [DSC$K_CLASS_P]:
1321 1449 BEGIN
1322 1450 DBG$PRINT(UPLIT BYTE(%ASCIC'procedure descriptor type,'), 0);
1323 1451 DBG$TRANSLATE_TYPECODE(.DTYPE);
1324 1452 IF .LENGTH EQL 0
1325 1453 THEN
1326 1454     DBG$PRINT(UPLIT BYTE(%ASCIC', no function value is returned'), 0)
1327 1455 ELSE
1328 1456     DBG$PRINT(UPLIT BYTE(%ASCIC', function value size: !UL'),
1329 1457               .LENGTH);
1330 1458
1331 1459 END;
1332 1460
1333 1461 ! Reserved for use by the VAX Debugger.
1334 1462
1335 1463 [DSC$K_CLASS_J]:
1336 1464 BEGIN
1337 1465     DBG$PRINT(UPLIT BYTE(%ASCIC'label descriptor type'), 0);
1338 1466 END;
1339 1467
1340 1468 ! Decimal Scalar String Descriptor, is used to describe decimal size
1341 1469 ! and scaling information for scalar data and simple strings.
1342 1470
1343 1471 [DSC$K_CLASS_SD]:
1344 1472 BEGIN
1345 1473     IF .VAX_DESCR[DSC$V_FL_BINSIZE]
1346 1474     THEN
1347 1475         DBG$PRINT(UPLIT BYTE(%ASCIC'binary scalar string descriptor type,'), 0)
1348 1476     ELSE
1349 1477         DBG$PRINT(UPLIT BYTE(%ASCIC'decimal scalar string descriptor type,'), 0);
1350 1478     DBG$TRANSLATE_TYPECODE(.DTYPE);
1351 1479     DTYPE_LENGTH(LENGTH);
1352 1480     IF NOT ((.LENGTH EQL .VAX_DESCR[DSC$B_DIGITS]) AND
1353 1481            (.VAX_DESCR[DSC$B_DTYPE] EQL DSC$K_DTYPE_P))
1354 1482     THEN
1355 1483         DTYPE_DIGITS(VAX_DESCR[DSC$B_DIGITS]);
1356 1484         DTYPE_SCALE(VAX_DESCR[DSC$B_SCALE]);
1357 1485     END;
1358 1486
1359 1487 ! Varying String Descriptor, is used for varying strings.
1360 1488
1361 1489 [DSC$K_CLASS_VS]:
1362 1490 BEGIN
1363 1491     DBG$PRINT(UPLIT BYTE(%ASCIC'varying string descriptor type,'), 0);
1364 1492     DBG$TRANSLATE_TYPECODE(.DTYPE);
1365 1493     DBG$PRINT(UPLIT BYTE(%ASCIC', size: !UL bytes'), .VAX_DESCR[DSC$W_MAXSTRLEN]);
1366 1494 END;
1367 1495
1368 1496 ! Varying String Array Descriptor, is used to specified an array
1369 1497 ! of varying strings.
1370 1498
1371 1499
1372 1500
1373 1501
1374 1502
```



```
: 1375      1503      2      [DSC$K_CLASS_VSA]:
: 1376      1504      2      BEGIN
: 1377      1505      2      DBG$PRINT(UPLIT BYTE(%ASCIC 'varying string array descriptor type, '), 0);
: 1378      1506      2      DBG$TRANSLATE TYPECODE(.DTYPE);
: 1379      1507      2      DBG$PRINT(UPLIT BYTE(%ASCIC ', size: !UL bytes'), .VAX_DESCR[DSC$W_MAXSTRLEN]);
: 1380      1508      2      END;
: 1381      1509      2
: 1382      1510      2
: 1383      1511      2      ! Unaligned Bit String Descriptor, is used to pass an unaligned bit
: 1384      1512      2      string.
: 1385      1513      2
: 1386      1514      2      [DSC$K_CLASS_UBS]:
: 1387      1515      2      BEGIN
: 1388      1516      2      DBG$PRINT(UPLIT BYTE(%ASCIC 'unaligned bit string descriptor type, '), 0);
: 1389      1517      2      DBG$TRANSLATE TYPECODE(.DTYPE);
: 1390      1518      2      DBG$PRINT(UPLIT BYTE(%ASCIC ', size: !UL bits'), .VAX_DESCR[DSC$W_LENGTH]);
: 1391      1519      2      END;
: 1392      1520      2
: 1393      1521      2
: 1394      1522      2
: 1395      1523      2      [INRANGE, OUTRANGE]:
: 1396      1524      2      BEGIN
: 1397      1525      2      DBG$PRINT(UPLIT BYTE(%ASCIC 'undefined descriptor type'), 0);
: 1398      1526      2      END;
: 1399      1527      2      TES;
: 1400      1528      2
: 1401      1529      2      RETURN 0;
: 1402      1530      1      END;
```

```
.PSECT DBG$PLIT,NOWRT, SHR, PIC,0
70 69 72 63 73 65 64 20 67 6E 69 72 74 73 17 00026 P.AAB: .ASCII <23>\string descriptor type,\
70 69 72 63 73 65 64 20 67 70 79 74 20 72 6F 74 00035
70 69 72 63 73 65 64 20 67 70 79 74 20 63 73 17 0003E P.AAC: .ASCII <23>\scalar descriptor type,\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 0004D
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 00056 P.AAD: .ASCII <7>\, size:\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 0005E P.AAE: .ASCII <9>\ !UL bits\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 00068 P.AAF: .ASCII <11>\ !UL digits\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 00074 P.AAG: .ASCII <10>\ !UL bytes\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 0007F P.AAH: .ASCII <31>\dynamic string descriptor type,\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 0008E
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 0009D
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 0009F P.AAI: .ASCII <7>\, size:\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000A7 P.AAJ: .ASCII <9>\ !UL bits\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000B1 P.AAK: .ASCII <11>\ !UL digits\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000BD P.AAL: .ASCII <10>\ !UL bytes\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000C8 P.AAM: .ASCII <22>\array descriptor type,\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000D7
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000DF P.AAN: .ASCII \$noncontiguous array descriptor type,\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000EE
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 000FD
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 00104 P.AAO: .ASCII \$unaligned bit array descriptor type,\
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 00113
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 00122
70 69 72 63 73 65 64 20 67 70 79 74 20 6F 74 00129 P.AAP: .ASCII <16>\ !UL dimensions,\
```



|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |        |        |                                           |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|--------|--------|-------------------------------------------|
| 63 | 73 | 65 | 64 | 20 | 65 | 72 | 75 | 64 | 65 | 63 | 6F | 72 | 70 | 1A | 00157 | P.AAT: | .ASCII | <26>\procedure descriptor type,\          |
| 20 | 6E | 6F | 69 | 74 | 63 | 6E | 75 | 66 | 20 | 6F | 6E | 20 | 2C | 1F | 00172 | P.AAU: | .ASCII | <31>\, no function value is returned\     |
| 6E | 72 | 75 | 74 | 65 | 72 | 20 | 73 | 69 | 20 | 65 | 75 | 6C | 61 | 76 | 00181 |        |        |                                           |
| 6C | 61 | 76 | 20 | 6E | 6F | 69 | 74 | 63 | 6E | 75 | 66 | 20 | 2C | 1A | 00192 | P.AAV: | .ASCII | <26>\, function value size: !UL\          |
| 74 | 70 | 69 | 72 | 63 | 73 | 65 | 64 | 20 | 6C | 65 | 62 | 61 | 6C | 15 | 001A1 |        |        |                                           |
| 20 | 72 | 61 | 6C | 61 | 63 | 73 | 20 | 79 | 72 | 61 | 6E | 69 | 62 | 25 | 001AD | P.AAW: | .ASCII | <21>\label descriptor type\               |
| 74 | 70 | 69 | 72 | 63 | 73 | 65 | 64 | 20 | 67 | 6E | 69 | 72 | 74 | 73 | 001BC | P.AAX: | .ASCII | \%binary scalar string descriptor type,\  |
| 72 | 61 | 6C | 61 | 63 | 73 | 20 | 6C | 61 | 6D | 69 | 63 | 65 | 64 | 26 | 001C3 |        |        |                                           |
| 70 | 69 | 72 | 63 | 73 | 65 | 64 | 20 | 67 | 6E | 69 | 72 | 74 | 73 | 20 | 001D2 | P.AAY: | .ASCII | \&decimal scalar string descriptor type,\ |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 001E1 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 001E9 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 001F8 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00207 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00210 | P.AAZ: | .ASCII | <7>\, size:\                              |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00218 | P.ABA: | .ASCII | <9>\ !UL bits\                            |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00222 | P.ABB: | .ASCII | <11>\ !UL digits\                         |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 0022E | P.ABC: | .ASCII | <10>\ !UL bytes\                          |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00239 | P.ABD: | .ASCII | <16>\ with !UL digits\                    |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00248 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 0024A | P.ABE: | .ASCII | <11>\ scaled !SL\                         |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00256 | P.ABF: | .ASCII | <31>\varying string descriptor type,\     |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00265 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00274 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00276 | P.ABG: | .ASCII | <17>\, size: !UL bytes\                   |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00285 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00288 | P.ABH: | .ASCII | \%varying string array descriptor type,\  |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00297 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002A6 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002AE | P.ABI: | .ASCII | <17>\, size: !UL bytes\                   |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002BD |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002C0 | P.ABJ: | .ASCII | \%unaligned bit string descriptor type,\  |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002CF |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002DE |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002E6 | P.ABK: | .ASCII | <16>\, size: !UL bits\                    |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002F5 |        |        |                                           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 002F7 | P.ABL: | .ASCII | <25>\undefined descriptor type\           |
|    |    |    |    |    |    | 2C | 65 | 70 | 79 | 74 | 20 | 72 | 6F | 74 | 00306 |        |        |                                           |

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

03FC 00000

|        |                                              |      |
|--------|----------------------------------------------|------|
| .ENTRY | DBG\$TRANSALATE_DESCR, Save R2,R3,R4,R5,R6,- | 1332 |
| MOVAB  | R7,R8,R9                                     |      |
| MOVAB  | DBG\$TRANSALATE_TYPECODE, R9                 |      |
| MOVAB  | DBG\$PRINT, R8                               |      |
| MOVAB  | P.ABL, R7                                    |      |
| MOVL   | VAX \$DESCR, R4                              | 1356 |
| MOVZBL | 2(R4), DTYPE                                 |      |
| MOVZWL | (R4), LENGTH                                 | 1357 |
| MOVZBL | 3(R4), R2                                    | 1358 |

|    |           |    |    |       |
|----|-----------|----|----|-------|
| 59 | 0000V     | CF | 9E | 00002 |
| 58 | 00000000G | 00 | 9E | 00007 |
| 57 | 00000000' | EF | 9E | 0000E |
| 54 | 04        | AC | D0 | 00015 |
| 53 | 02        | A4 | 9A | 00019 |
| 55 |           | 64 | 3C | 0001D |
| 52 | 03        | A4 | 9A | 00020 |



|      |      |      |      |    |       |            |                             |      |
|------|------|------|------|----|-------|------------|-----------------------------|------|
| 00BB | 0D   | 01   | 52   | 8F | 00024 | CASEB      | R2, #1, #13                 |      |
| 001C | 001C | 007C | 0023 |    | 00028 | .WORD      | 3\$-1\$,-                   |      |
| 01ED | 0156 | 001C | 0134 |    | 00030 |            | 10\$-1\$,-                  |      |
|      | 01D6 | 00BB | 015E |    | 00038 |            | 2\$-1\$,-                   |      |
|      |      | 00BB | 0202 |    | 00040 |            | 17\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 26\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 2\$-1\$,-                   |      |
|      |      |      |      |    |       |            | 28\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 2\$-1\$,-                   |      |
|      |      |      |      |    |       |            | 30\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 17\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 40\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 41\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 42\$-1\$,-                  |      |
|      |      |      |      |    |       |            | 17\$-1\$                    |      |
|      |      |      | 7E   | D4 | 00044 | 2\$: CLRL  | -(SP)                       | 1524 |
|      |      |      | 57   | DD | 00046 | PUSHL      | R7                          |      |
|      |      |      | 0080 | 31 | 00048 | BRW        | 13\$                        |      |
| 0E   |      |      | 53   | D1 | 0004B | 3\$: CMPL  | DTYPE, #14                  | 1368 |
|      |      |      | 05   | 19 | 0004E | BLSS       | 4\$                         |      |
| 15   |      |      | 53   | D1 | 00050 | CMPL       | DTYPE, #21                  |      |
|      |      |      | 0A   | 15 | 00053 | BLEQ       | 5\$                         |      |
| 01   |      |      | 53   | D1 | 00055 | 4\$: CMPL  | DTYPE, #1                   | 1369 |
|      |      |      | 05   | 13 | 00058 | BEQL       | 5\$                         |      |
| 22   |      |      | 53   | D1 | 0005A | CMPL       | DTYPE, #34                  | 1370 |
|      |      |      | 08   | 12 | 0005D | BNEQ       | 6\$                         |      |
|      |      |      | 7E   | D4 | 0005F | 5\$: CLRL  | -(SP)                       | 1372 |
|      |      | FD2F | C7   | 9F | 00061 | PUSHAB     | P.AAB                       |      |
|      |      |      | 06   | 11 | 00065 | BRB        | 7\$                         |      |
|      |      |      | 7E   | D4 | 00067 | 6\$: CLRL  | -(SP)                       | 1374 |
|      |      | FD47 | C7   | 9F | 00069 | PUSHAB     | P.AAC                       |      |
| 68   |      |      | 02   | FB | 0006D | 7\$: CALLS | #2, DBG\$PRINT              |      |
|      |      |      | 53   | DD | 00070 | PUSHL      | DTYPE                       | 1376 |
| 69   |      |      | 01   | FB | 00072 | CALLS      | #1, DBG\$TRANSLATE_TYPECODE |      |
|      |      |      | 55   | D5 | 00075 | TSTL       | LENGTH                      | 1377 |
|      |      |      | 3B   | 13 | 00077 | BEQL       | 11\$                        |      |
|      |      |      | 7E   | D4 | 00079 | CLRL       | -(SP)                       |      |
|      |      | FD5F | C7   | 9F | 0007B | PUSHAB     | P.AAD                       |      |
| 68   |      |      | 02   | FB | 0007F | CALLS      | #2, DBG\$PRINT              |      |
| 01   |      |      | 53   | D1 | 00082 | CMPL       | DTYPE, #1                   |      |
|      |      |      | 08   | 12 | 00085 | BNEQ       | 8\$                         |      |
|      |      |      | 55   | DD | 00087 | PUSHL      | LENGTH                      |      |
|      |      | FD67 | C7   | 9F | 00089 | PUSHAB     | P.AAE                       |      |
|      |      |      | 51   | 11 | 0008D | BRB        | 16\$                        |      |
| 15   |      |      | 53   | D1 | 0008F | 8\$: CMPL  | DTYPE, #21                  |      |
|      |      |      | 08   | 12 | 00092 | BNEQ       | 9\$                         |      |
|      |      |      | 55   | DD | 00094 | PUSHL      | LENGTH                      |      |
|      |      | FD71 | C7   | 9F | 00096 | PUSHAB     | P.AAF                       |      |
|      |      |      | 44   | 11 | 0009A | BRB        | 16\$                        |      |
|      |      |      | 55   | DD | 0009C | 9\$: PUSHL | LENGTH                      |      |
|      |      | FD7D | C7   | 9F | 0009E | PUSHAB     | P.AAG                       |      |
|      |      |      | 3C   | 11 | 000A2 | BRB        | 16\$                        |      |
|      |      |      | 7E   | D4 | 000A4 | 10\$: CLRL | -(SP)                       | 1385 |
|      |      | FD88 | C7   | 9F | 000A6 | PUSHAB     | P.AAH                       |      |
| 68   |      |      | 02   | FB | 000AA | CALLS      | #2, DBG\$PRINT              |      |
|      |      |      | 53   | DD | 000AD | PUSHL      | DTYPE                       | 1386 |
| 69   |      |      | 01   | FB | 000AF | CALLS      | #1, DBG\$TRANSLATE_TYPECODE |      |



|    |         |      |       |       |       |        |                              |      |
|----|---------|------|-------|-------|-------|--------|------------------------------|------|
|    |         | 55   | D5    | 000B2 |       | TSTL   | LENGTH                       | 1387 |
|    |         | 01   | 12    | 000B4 | 11\$: | BNEQ   | 12\$                         |      |
|    |         |      | 04    | 000B6 |       | RET    |                              |      |
|    | FDA8    | 7E   | D4    | 000B7 | 12\$: | CLRL   | -(SP)                        |      |
| 68 |         | C7   | 9F    | 000B9 |       | PUSHAB | P.AAI                        |      |
| 01 |         | 02   | FB    | 000BD |       | CALLS  | #2, DBG\$PRINT               |      |
|    |         | 53   | D1    | 000C0 |       | CMPL   | DTYPE, #1                    |      |
|    |         | 08   | 12    | 000C3 |       | BNEQ   | 14\$                         |      |
|    | FDB0    | 55   | DD    | 000C5 |       | PUSHL  | LENGTH                       |      |
|    |         | C7   | 9F    | 000C7 |       | PUSHAB | P.AAJ                        |      |
|    |         | 13   | 11    | 000CB | 13\$: | BRB    | 16\$                         |      |
| 15 |         | 53   | D1    | 000CD | 14\$: | CMPL   | DTYPE, #21                   |      |
|    |         | 08   | 12    | 000D0 |       | BNEQ   | 15\$                         |      |
|    | FDBA    | 55   | DD    | 000D2 |       | PUSHL  | LENGTH                       |      |
|    |         | C7   | 9F    | 000D4 |       | PUSHAB | P.AAK                        |      |
|    |         | 06   | 11    | 000D8 |       | BRB    | 16\$                         |      |
|    | FDC6    | 55   | DD    | 000DA | 15\$: | PUSHL  | LENGTH                       |      |
|    |         | C7   | 9F    | 000DC |       | PUSHAB | P.AAL                        |      |
|    |         | 015A | 31    | 000E0 | 16\$: | BRW    | 43\$                         |      |
| 04 |         | 52   | 91    | 000E3 | 17\$: | CMPB   | R2, #4                       | 1402 |
|    |         | 08   | 12    | 000E6 |       | BNEQ   | 18\$                         |      |
|    | FDD1    | 7E   | D4    | 000E8 |       | CLRL   | -(SP)                        | 1403 |
|    |         | C7   | 9F    | 000EA |       | PUSHAB | P.AAM                        |      |
|    |         | 18   | 11    | 000EE |       | BRB    | 20\$                         |      |
| 0A |         | 52   | 91    | 000F0 | 18\$: | CMPB   | R2, #10                      | 1410 |
|    |         | 08   | 12    | 000F3 |       | BNEQ   | 19\$                         |      |
|    | FDE8    | 7E   | D4    | 000F5 |       | CLRL   | -(SP)                        | 1411 |
|    |         | C7   | 9F    | 000F7 |       | PUSHAB | P.AAN                        |      |
|    |         | 0B   | 11    | 000FB |       | BRB    | 20\$                         |      |
| 0E |         | 52   | 91    | 000FD | 19\$: | CMPB   | R2, #14                      | 1417 |
|    |         | 09   | 12    | 00100 |       | BNEQ   | 21\$                         |      |
|    | FE0D    | 7E   | D4    | 00102 |       | CLRL   | -(SP)                        | 1418 |
|    |         | C7   | 9F    | 00104 |       | PUSHAB | P.AAO                        |      |
| 68 |         | 02   | FB    | 00108 | 20\$: | CALLS  | #2, DBG\$PRINT               |      |
| 56 | 0B      | A4   | 9A    | 0010B | 21\$: | MOVZBL | 11(R4), R6                   | 1421 |
|    |         | 56   | DD    | 0010F |       | PUSHL  | R6                           |      |
|    | FE32    | C7   | 9F    | 00111 |       | PUSHAB | P.AAP                        |      |
|    |         | 02   | FB    | 00115 |       | CALLS  | #2, DBG\$PRINT               |      |
| 68 |         | 7E   | D4    | 00118 |       | CLRL   | -(SP)                        | 1426 |
|    | FE43    | C7   | 9F    | 0011A |       | PUSHAB | P.AAQ                        |      |
|    |         | 02   | FB    | 0011E |       | CALLS  | #2, DBG\$PRINT               |      |
| 68 |         | 02   | FB    | 0011E |       | CALLS  | #2, DBG\$PRINT               |      |
| 52 | 14      | A4   | 9E    | 00121 |       | MOVAB  | 20(R4), BOUNDS_PTR           | 1427 |
| 52 |         | 6246 | DE    | 00125 |       | MOVAL  | (BOUNDS_PTR)[R6], BOUNDS_PTR | 1428 |
|    |         | 53   | D4    | 00129 |       | CLRL   | BOUND_NO                     | 1435 |
|    |         | 2A   | 11    | 0012B |       | BRB    | 25\$                         |      |
| 51 |         | 01   | 78    | 0012D | 22\$: | ASHL   | #1, BOUND_NO, R1             |      |
| 50 |         | 01   | 78    | 00131 |       | ASHL   | #1, BOUND_NO, R0             | 1434 |
|    |         | 53   | D1    | 00135 |       | CMPL   | BOUND_NO, R6                 | 1431 |
|    |         | 0E   | 12    | 00138 |       | BNEQ   | 23\$                         |      |
|    | FC A241 | DD   | 0013A |       |       | PUSHL  | -4(BOUNDS_PTR)[R1]           | 1435 |
|    | F8 A240 | DD   | 0013E |       |       | PUSHL  | -8(BOUNDS_PTR)[R0]           | 1434 |
|    | FE4E    | C7   | 9F    | 00142 |       | PUSHAB | P.AAR                        | 1433 |
|    |         | 0C   | 11    | 00146 |       | BRB    | 24\$                         |      |
|    | FC A241 | DD   | 00148 | 23\$: |       | PUSHL  | -4(BOUNDS_PTR)[R1]           | 1439 |
|    | F8 A240 | DD   | 0014C |       |       | PUSHL  | -8(BOUNDS_PTR)[R0]           | 1438 |
|    | FE57    | C7   | 9F    | 00150 |       | PUSHAB | P.AAS                        | 1437 |
| 68 |         | 03   | FB    | 00154 | 24\$: | CALLS  | #3, DBG\$PRINT               |      |



|    |      |    |    |       |       |        |                              |                       |      |
|----|------|----|----|-------|-------|--------|------------------------------|-----------------------|------|
| D2 | 53   | 56 | F3 | 00157 | 25\$: | AOBLEQ | R6, BOUND_NO, 22\$           | 1429                  |      |
|    |      |    | 04 | 0015B |       | RET    |                              | 1358                  |      |
|    |      | 7E | D4 | 0015C | 26\$: | CLRL   | -(SP)                        | 1450                  |      |
|    | FE60 | C7 | 9F | 0015E |       | PUSHAB | P.AAT                        |                       |      |
|    | 68   | 02 | FB | 00162 |       | CALLS  | #2, DBG\$PRINT               |                       |      |
|    |      | 53 | DD | 00165 |       | PUSHL  | DTYPE                        | 1451                  |      |
|    | 69   | 01 | FB | 00167 |       | CALLS  | #1, DBG\$TRANSALATE_TYPECODE |                       |      |
|    |      | 55 | D5 | 0016A |       | TSTL   | LENGTH                       | 1452                  |      |
|    |      | 08 | 12 | 0016C |       | BNEQ   | 27\$                         |                       |      |
|    |      | 7E | D4 | 0016E |       | CLRL   | -(SP)                        | 1454                  |      |
|    | FE7B | C7 | 9F | 00170 |       | PUSHAB | P.AAU                        |                       |      |
|    |      | 0E | 11 | 00174 |       | BRB    | 29\$                         |                       |      |
|    |      | 55 | DD | 00176 | 27\$: | PUSHL  | LENGTH                       | 1457                  |      |
|    | FE9B | C7 | 9F | 00178 |       | PUSHAB | P.AAV                        | 1456                  |      |
|    |      | 7E | 11 | 0017C |       | BRB    | 39\$                         |                       |      |
|    |      | 7E | D4 | 0017E | 28\$: | CLRL   | -(SP)                        | 1466                  |      |
|    | FEB6 | C7 | 9F | 00180 |       | PUSHAB | P.AAW                        |                       |      |
|    |      | 76 | 11 | 00184 | 29\$: | BRB    | 39\$                         |                       |      |
| 08 | 0A   | A4 | 03 | E1    | 00186 | 30\$:  | BBC                          | #3, 10(R4), 31\$      | 1475 |
|    |      | 7E | D4 | 0018B |       | CLRL   | -(SP)                        | 1477                  |      |
|    | FECC | C7 | 9F | 0018D |       | PUSHAB | P.AAX                        |                       |      |
|    |      | 06 | 11 | 00191 |       | BRB    | 32\$                         |                       |      |
|    |      | 7E | D4 | 00193 | 31\$: | CLRL   | -(SP)                        | 1479                  |      |
|    | FEF2 | C7 | 9F | 00195 |       | PUSHAB | P.AAY                        |                       |      |
|    | 68   | 02 | FB | 00199 | 32\$: | CALLS  | #2, DBG\$PRINT               |                       |      |
|    |      | 53 | DD | 0019C |       | PUSHL  | DTYPE                        | 1480                  |      |
|    | 69   | 01 | FB | 0019E |       | CALLS  | #1, DBG\$TRANSALATE_TYPECODE |                       |      |
|    |      | 55 | D5 | 001A1 |       | TSTL   | LENGTH                       | 1481                  |      |
|    |      | 2C | 13 | 001A3 |       | BEQL   | 36\$                         |                       |      |
|    |      | 7E | D4 | 001A5 |       | CLRL   | -(SP)                        |                       |      |
|    | FF19 | C7 | 9F | 001A7 |       | PUSHAB | P.AAZ                        |                       |      |
|    | 68   | 02 | FB | 001AB |       | CALLS  | #2, DBG\$PRINT               |                       |      |
|    | 01   | 53 | D1 | 001AE |       | CMPL   | DTYPE, #1                    |                       |      |
|    |      | 08 | 12 | 001B1 |       | BNEQ   | 33\$                         |                       |      |
|    |      | 55 | DD | 001B3 |       | PUSHL  | LENGTH                       |                       |      |
|    | FF21 | C7 | 9F | 001B5 |       | PUSHAB | P.ABA                        |                       |      |
|    |      | 13 | 11 | 001B9 |       | BRB    | 35\$                         |                       |      |
|    | 15   | 53 | D1 | 001BB | 33\$: | CMPL   | DTYPE, #21                   |                       |      |
|    |      | 08 | 12 | 001BE |       | BNEQ   | 34\$                         |                       |      |
|    |      | 55 | DD | 001C0 |       | PUSHL  | LENGTH                       |                       |      |
|    | FF2B | C7 | 9F | 001C2 |       | PUSHAB | P.ABB                        |                       |      |
|    |      | 06 | 11 | 001C6 |       | BRB    | 35\$                         |                       |      |
|    |      | 55 | DD | 001C8 | 34\$: | PUSHL  | LENGTH                       |                       |      |
|    | FF37 | C7 | 9F | 001CA |       | PUSHAB | P.ABC                        |                       |      |
|    | 68   | 02 | FB | 001CE | 35\$: | CALLS  | #2, DBG\$PRINT               |                       |      |
| 55 | 09   | A4 | 00 | ED    | 001D1 | 36\$:  | CMPZV                        | #0, #8, 9(R4), LENGTH | 1482 |
|    |      | 06 | 12 | 001D7 |       | BNEQ   | 37\$                         |                       |      |
|    | 15   | 02 | A4 | 91    | 001D9 |        | CMPB                         | 2(R4), #21            | 1483 |
|    |      | 10 | 13 | 001DD |       | BEQL   | 38\$                         |                       |      |
|    |      | 09 | A4 | 95    | 001DF | 37\$:  | TSTB                         | 9(R4)                 | 1485 |
|    |      | 0B | 13 | 001E2 |       | BEQL   | 38\$                         |                       |      |
|    | 7E   | 09 | A4 | 9A    | 001E4 |        | MOVZBL                       | 9(R4), -(SP)          |      |
|    | FF42 | C7 | 9F | 001E8 |       | PUSHAB | P.ABD                        |                       |      |
|    | 68   | 02 | FB | 001EC |       | CALLS  | #2, DBG\$PRINT               |                       |      |
|    |      | 08 | A4 | 95    | 001EF | 38\$:  | TSTB                         | 8(R4)                 | 1486 |
|    |      | 4C | 13 | 001F2 |       | BEQL   | 44\$                         |                       |      |
|    | 7E   | 08 | A4 | 98    | 001F4 |        | CVTBL                        | 8(R4), -(SP)          |      |



|    |      |    |       |       |        |                             |        |
|----|------|----|-------|-------|--------|-----------------------------|--------|
|    | FF53 | C7 | 9F    | 001F8 | PUSHAB | P.ABE                       | :      |
|    |      | 3F | 11    | 001FC | BRB    | 43\$                        | :      |
|    |      | 7E | D4    | 001FE | CLRL   | -(SP)                       | : 1494 |
| 68 | FF5F | C7 | 9F    | 00200 | PUSHAB | P.ABF                       | :      |
|    |      | 02 | FB    | 00204 | CALLS  | #2, DBG\$PRINT              | :      |
|    |      | 53 | DD    | 00207 | PUSHL  | DTYPE                       | : 1495 |
| 69 |      | 01 | FB    | 00209 | CALLS  | #1, DBG\$TRANSLATE_TYPECODE | :      |
| 7E |      | 64 | 3C    | 0020C | MOVZWL | (R4), -(SP)                 | : 1496 |
|    | FF7F | C7 | 9F    | 0020F | PUSHAB | P.ABG                       | :      |
|    |      | 28 | 11    | 00213 | BRB    | 43\$                        | :      |
|    |      | 7E | D4    | 00215 | CLRL   | -(SP)                       | : 1505 |
|    | 91   | A7 | 9F    | 00217 | PUSHAB | P.ABH                       | :      |
| 68 |      | 02 | FB    | 0021A | CALLS  | #2, DBG\$PRINT              | :      |
|    |      | 53 | DD    | 0021D | PUSHL  | DTYPE                       | : 1506 |
| 69 |      | 01 | FB    | 0021F | CALLS  | #1, DBG\$TRANSLATE_TYPECODE | :      |
| 7E |      | 64 | 3C    | 00222 | MOVZWL | (R4), -(SP)                 | : 1507 |
|    | B7   | A7 | 9F    | 00225 | PUSHAB | P.ABI                       | :      |
|    |      | 13 | 11    | 00228 | BRB    | 43\$                        | :      |
|    |      | 7E | D4    | 0022A | CLRL   | -(SP)                       | : 1516 |
|    | C9   | A7 | 9F    | 0022C | PUSHAB | P.ABJ                       | :      |
| 68 |      | 02 | FB    | 0022F | CALLS  | #2, DBG\$PRINT              | :      |
|    |      | 53 | DD    | 00232 | PUSHL  | DTYPE                       | : 1517 |
| 69 |      | 01 | FB    | 00234 | CALLS  | #1, DBG\$TRANSLATE_TYPECODE | :      |
| 7E |      | 64 | 3C    | 00237 | MOVZWL | (R4), -(SP)                 | : 1518 |
|    | EF   | A7 | 9F    | 0023A | PUSHAB | P.ABK                       | :      |
| 68 |      | 02 | FB    | 0023D | CALLS  | #2, DBG\$PRINT              | :      |
|    |      | 04 | 00240 | 44\$: | RET    |                             | : 1530 |

; Routine Size: 577 bytes, Routine Base: DBG\$CODE + 063B



```
: 1404      1531 1 GLOBAL ROUTINE DBG$TRANSLATE_KIND(RSTPTR): =
: 1405      1532 1
: 1406      1533 1 FUNCTION
: 1407      1534 1     Given RST kind field, transalte it into readable format, also
: 1408      1535 1     return the length of the RST entry.
: 1409      1536 1
: 1410      1537 1 INPUT
: 1411      1538 1     KIND - Symbol's RST kind field.
: 1412      1539 1
: 1413      1540 1 OUTPUTS
: 1414      1541 1     The length of the RST entry is returned.
: 1415      1542 1
: 1416      1543 2 BEGIN
: 1417      1544 2
: 1418      1545 2 MAP
: 1419      1546 2     RSTPTR: REF RST$ENTRY;           ! Pointer to Symbol's RST entry
: 1420      1547 2
: 1421      1548 2 LOCAL
: 1422      1549 2     LENGTH;                       ! The length of the RST entry
: 1423      1550 2
: 1424      1551 2
: 1425      1552 2 CASE .RSTPTR[RST$B_KIND] FROM RST$K_KIND_MINIMUM TO RST$K_KIND_MAXIMUM OF
: 1426      1553 2 SET
: 1427      1554 2
: 1428      1555 2     [RST$K_MODULE]:
: 1429      1556 2         BEGIN
: 1430      1557 2             DBG$PRINT(UPLIT BYTE(%ASCIC 'module'), 0);
: 1431      1558 2             LENGTH = RST$K_MODENTSIZ;
: 1432      1559 2             END;
: 1433      1560 2
: 1434      1561 2     [RST$K_ROUTINE]:
: 1435      1562 2         BEGIN
: 1436      1563 2             DBG$PRINT(UPLIT BYTE(%ASCIC 'routine'), 0);
: 1437      1564 2             LENGTH = RST$K_ROUTENTSIZ;
: 1438      1565 2             END;
: 1439      1566 2
: 1440      1567 2     [RST$K_BLOCK]:
: 1441      1568 2         BEGIN
: 1442      1569 2             DBG$PRINT(UPLIT BYTE(%ASCIC 'block'), 0);
: 1443      1570 2             LENGTH = RST$K_LEXENTSIZ;
: 1444      1571 2             END;
: 1445      1572 2
: 1446      1573 2     [RST$K_ENTRY]:
: 1447      1574 2         BEGIN
: 1448      1575 2             DBG$PRINT(UPLIT BYTE(%ASCIC 'entry'), 0);
: 1449      1576 2             LENGTH = RST$K_EPTENTSIZ;
: 1450      1577 2             END;
: 1451      1578 2
: 1452      1579 2     [RST$K_LABEL]:
: 1453      1580 2         BEGIN
: 1454      1581 2             DBG$PRINT(UPLIT BYTE(%ASCIC 'label'), 0);
: 1455      1582 2             LENGTH = RST$K_LBLENTSIZ;
: 1456      1583 2             END;
: 1457      1584 2
: 1458      1585 2     [RST$K_LINE]:
: 1459      1586 2         BEGIN
: 1460      1587 2             DBG$PRINT(UPLIT BYTE(%ASCIC 'line'), 0);
```



```
: 1461      1588      3
: 1462      1589      2
: 1463      1590      2
: 1464      1591      2
: 1465      1592      2
: 1466      1593      2
: 1467      1594      2
: 1468      1595      2
: 1469      1596      2
: 1470      1597      2
: 1471      1598      2
: 1472      1599      2
: 1473      1600      2
: 1474      1601      2
: 1475      1602      2
: 1476      1603      2
: 1477      1604      2
: 1478      1605      2
: 1479      1606      2
: 1480      1607      2
: 1481      1608      2
: 1482      1609      2
: 1483      1610      2
: 1484      1611      2
: 1485      1612      2
: 1486      1613      2
: 1487      1614      2
: 1488      1615      2
: 1489      1616      2
: 1490      1617      2
: 1491      1618      2
: 1492      1619      2
: 1493      1620      2
: 1494      1621      2
: 1495      1622      2
: 1496      1623      2
: 1497      1624      2
: 1498      1625      2
: 1499      1626      2
: 1500      1627      2
: 1501      1628      2
: 1502      1629      2
: 1503      1630      2
: 1504      1631      2
: 1505      1632      2
: 1506      1633      2
: 1507      1634      2
: 1508      1635      2
: 1509      1636      2
: 1510      1637      2
: 1511      1638      2
: 1512      1639      2
: 1513      1640      2
: 1514      1641      2
: 1515      1642      2
: 1516      1643      2
: 1517      1644      3
```

```
    LENGTH = RST$K_LINENTSIZ;
    END;

[RST$K_DATA]:
    BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC 'data'), 0);
    LENGTH = RST$K_DATENTSIZ;
    END;

[RST$K_TYPCOMP]:
    BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC 'record component'), 0);
    LENGTH = RST$K_DATENTSIZ;
    END;

! You really should not get this kind.

[RST$K_TYPE]:
    BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC 'type'), 0);
    LENGTH = RST$K_TYPCNTSIZ + .RSTPTR[RST$L_TYPCOMPENT];
    END;

! You can get this kind only if /TYPE is specified. Routine
! GET_VARIANT is called.

[RST$K_VARIANT]:
    BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC 'record variants'), 0);
    LENGTH = RST$K_VARENTSIZ + .RSTPTR[RST$L_VARETCNT];
    END;

! You really should not get this kind.

[RST$K_INVOCNUM]:
    BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC 'invocnum'), 0);
    LENGTH = RST$K_INVENTSIZ;
    END;

! You really should not get this kind.

[RST$K_OVERLOAD]:
    BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC 'overloaded symbol'), 0);
    LENGTH = RST$K_OLENTSIZ;
    END;

! Any other kind is invalid.

[INRANGE, OUTRANGE]:
    BEGIN
```



```
: 1518      1645      3      DBG$PRINT(UPLIT BYTE(%ASCIC 'invalid kind'), 0);
: 1519      1646      3      LENGTH = 0;
: 1520      1647      2      END;
: 1521      1648      2
: 1522      1649      2      TES;
: 1523      1650      2      RETURN .LENGTH;
: 1524      1651      2
: 1525      1652      2      END;
: 1526      1653      1
```

```
                                .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0
                                65  65  6C  75  64  6F  6D  06  00311 P.ABM:  .ASCII  <6>\module\
                                65  6E  69  74  75  6F  72  07  00318 P.ABN:  .ASCII  <7>\routine\
                                68  63  6F  6C  62  05  00320 P.ABO:  .ASCII  <5>\block\
                                79  72  74  6E  65  05  00326 P.ABP:  .ASCII  <5>\entry\
                                6C  65  62  61  6C  05  0032C P.ABQ:  .ASCII  <5>\label\
                                65  6E  69  6C  04  00332 P.ABR:  .ASCII  <4>\line\
                                61  74  61  64  04  00337 P.ABS:  .ASCII  <4>\data\
65  6E  6F  70  6D  6F  63  20  64  72  6F  63  65  72  10  0033C P.ABT:  .ASCII  <16>\record component\
                                74  6E  0034B
                                74  6E  61  69  72  61  76  20  64  72  6F  63  65  72  0F  0034D P.ABU:  .ASCII  <4>\type\
                                73  00352 P.ABV:  .ASCII  <15>\record variants\
                                73  00361
6D  79  73  20  64  65  6D  75  6E  63  6F  76  6E  69  08  00362 P.ABW:  .ASCII  <8>\invocnum\
                                64  61  6F  6C  72  65  76  6F  11  0036B P.ABX:  .ASCII  <17>\overloaded symbol\
                                64  6E  69  6B  20  64  69  6C  61  76  6E  69  0C  0037A
                                64  6E  69  6B  20  64  69  6C  61  76  6E  69  0C  0037D P.ABY:  .ASCII  <12>\invalid kind\
```

```
                                .PSECT  DBG$CODE,NOWRT,  SHR,  PIC,0
                                54  00000000G  00  001C  00000
                                53  00000000'  EF  9E  00002
                                52  04  AC  D0  00010
                                00  14  A2  8F  00014
0039      0D      0026      001C      00019 1$:
006D      0032      0053      004C      00021
007B      005F      001C      0040      00029
                                0095      0089      00031
                                7E  D4  00035 2$:
                                53  DD  00037
                                02  FB  00039
                                64
                                .ENTRY  DBG$TRANSLATE_KIND, Save R2,R3,R4
                                MOVAB   DBG$PRINT, R4
                                MOVAB   P.ABY, R3
                                MOVL    RSTPTR, R2
                                CASEB   20(R2), #0, #13
                                .WORD   2$-1$, -
                                3$-1$, -
                                4$-1$, -
                                5$-1$, -
                                8$-1$, -
                                9$-1$, -
                                11$-1$, -
                                13$-1$, -
                                6$-1$, -
                                2$-1$, -
                                12$-1$, -
                                14$-1$, -
                                15$-1$, -
                                17$-1$, -
                                CLRL    -(SP)
                                PUSHL   R3
                                CALLS   #2, DBG$PRINT
```

1531

1552

1645



|  |    |    |    |    |             |        |                     |   |      |
|--|----|----|----|----|-------------|--------|---------------------|---|------|
|  |    |    | 50 | D4 | 0003C       | CLRL   | LENGTH              | : | 1646 |
|  |    |    |    | 04 | 0003E       | RET    |                     | : | 1552 |
|  |    |    | 7E | D4 | 0003F 3\$:  | CLRL   | -(SP)               | : | 1557 |
|  |    |    | A3 | 9F | 00041       | PUSHAB | P.ABM               | : |      |
|  | 64 |    | 02 | FB | 00044       | CALLS  | #2, DBG\$PRINT      | : |      |
|  | 50 |    | 0C | D0 | 00047       | MOVL   | #12, LENGTH         | : | 1558 |
|  |    |    |    | 04 | 0004A       | RET    |                     | : | 1552 |
|  |    |    | 7E | D4 | 0004B 4\$:  | CLRL   | -(SP)               | : | 1563 |
|  |    |    | A3 | 9F | 0004D       | PUSHAB | P.ABN               | : |      |
|  |    |    | 0C | 11 | 00050       | BRB    | 7\$                 | : |      |
|  |    |    | 7E | D4 | 00052 5\$:  | CLRL   | -(SP)               | : | 1569 |
|  |    |    | A3 | 9F | 00054       | PUSHAB | P.ABO               | : |      |
|  |    |    | 18 | 11 | 00057       | BRB    | 10\$                | : |      |
|  |    |    | 7E | D4 | 00059 6\$:  | CLRL   | -(SP)               | : | 1575 |
|  |    |    | A9 | 9F | 0005B       | PUSHAB | P.ABP               | : |      |
|  | 64 |    | 02 | FB | 0005E 7\$:  | CALLS  | #2, DBG\$PRINT      | : |      |
|  | 50 |    | 0B | D0 | 00061       | MOVL   | #11, LENGTH         | : | 1576 |
|  |    |    |    | 04 | 00064       | RET    |                     | : | 1552 |
|  |    |    | 7E | D4 | 00065 8\$:  | CLRL   | -(SP)               | : | 1581 |
|  |    |    | A3 | 9F | 00067       | PUSHAB | P.ABQ               | : |      |
|  |    |    | 3B | 11 | 0006A       | BRB    | 16\$                | : |      |
|  |    |    | 7E | D4 | 0006C 9\$:  | CLRL   | -(SP)               | : | 1587 |
|  |    |    | A3 | 9F | 0006E       | PUSHAB | P.ABR               | : |      |
|  | 64 |    | 02 | FB | 00071 10\$: | CALLS  | #2, DBG\$PRINT      | : |      |
|  | 50 |    | 08 | D0 | 00074       | MOVL   | #8, LENGTH          | : | 1588 |
|  |    |    |    | 04 | 00077       | RET    |                     | : | 1552 |
|  |    |    | 7E | D4 | 00078 11\$: | CLRL   | -(SP)               | : | 1593 |
|  |    |    | A3 | 9F | 0007A       | PUSHAB | P.ABS               | : |      |
|  |    |    | 28 | 11 | 0007D       | BRB    | 16\$                | : |      |
|  |    |    | 7E | D4 | 0007F 12\$: | CLRL   | -(SP)               | : | 1599 |
|  |    |    | A3 | 9F | 00081       | PUSHAB | P.ABT               | : |      |
|  |    |    | 21 | 11 | 00084       | BRB    | 16\$                | : |      |
|  |    |    | 7E | D4 | 00086 13\$: | CLRL   | -(SP)               | : | 1608 |
|  |    |    | A3 | 9F | 00088       | PUSHAB | P.ABU               | : |      |
|  |    |    | 02 | FB | 0008B       | CALLS  | #2, DBG\$PRINT      | : |      |
|  | 50 | 28 | 0B | C1 | 0008E       | ADDL3  | #11, 40(R2), LENGTH | : | 1609 |
|  |    |    |    | 04 | 00093       | RET    |                     | : | 1552 |
|  |    |    | 7E | D4 | 00094 14\$: | CLRL   | -(SP)               | : | 1618 |
|  |    |    | A3 | 9F | 00096       | PUSHAB | P.ABV               | : |      |
|  |    |    | 02 | FB | 00099       | CALLS  | #2, DBG\$PRINT      | : |      |
|  | 50 | 08 | 06 | C1 | 0009C       | ADDL3  | #6, 8(R2), LENGTH   | : | 1619 |
|  |    |    |    | 04 | 000A1       | RET    |                     | : | 1552 |
|  |    |    | 7E | D4 | 000A2 15\$: | CLRL   | -(SP)               | : | 1627 |
|  |    |    | A3 | 9F | 000A4       | PUSHAB | P.ABW               | : |      |
|  |    |    | 02 | FB | 000A7 16\$: | CALLS  | #2, DBG\$PRINT      | : |      |
|  | 64 |    | 07 | D0 | 000AA       | MOVL   | #7, LENGTH          | : | 1628 |
|  | 50 |    |    | 04 | 000AD       | RET    |                     | : | 1552 |
|  |    |    | 7E | D4 | 000AE 17\$: | CLRL   | -(SP)               | : | 1636 |
|  |    |    | A3 | 9F | 000B0       | PUSHAB | P.ABX               | : |      |
|  |    |    | 02 | FB | 000B3       | CALLS  | #2, DBG\$PRINT      | : |      |
|  | 64 |    | 06 | D0 | 000B6       | MOVL   | #6, LENGTH          | : | 1637 |
|  | 50 |    |    | 04 | 000B9       | RET    |                     | : | 1653 |

; Routine Size: 186 bytes, Routine Base: DBG\$CODE + 087C



```
: 1528      1654 1 GLOBAL ROUTINE DBG$TRANSLATE_TYPECODE(TYPECODE): NOVALUE =
: 1529      1655 1
: 1530      1656 1 FUNCTION
: 1531      1657 1     Translate Data Type Code into user readable form.
: 1532      1658 1
: 1533      1659 1 INPUTS
: 1534      1660 1     TYPECODE - Data Type Code.
: 1535      1661 1
: 1536      1662 1 OUTPUTS
: 1537      1663 1     None.
: 1538      1664 1
: 1539      1665 1 BEGIN
: 1540      1666 2
: 1541      1667 2 CASE TYPECODE FROM DSC$K_DTYPE_LOWEST TO DSC$K_DTYPE_HIGHEST OF
: 1542      1668 2 SET
: 1543      1669 2
: 1544      1670 2     [DSC$K_DTYPE_V]:
: 1545      1671 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' bit field'), 0);
: 1546      1672 2
: 1547      1673 2     [DSC$K_DTYPE_BU]:
: 1548      1674 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' byte logical'), 0);
: 1549      1675 2
: 1550      1676 2     [DSC$K_DTYPE_WU]:
: 1551      1677 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' word logical'), 0);
: 1552      1678 2
: 1553      1679 2     [DSC$K_DTYPE_LU]:
: 1554      1680 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' longword logical'), 0);
: 1555      1681 2
: 1556      1682 2     [DSC$K_DTYPE_QU]:
: 1557      1683 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' quadword logical'), 0);
: 1558      1684 2
: 1559      1685 2     [DSC$K_DTYPE_B]:
: 1560      1686 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' byte integer'), 0);
: 1561      1687 2
: 1562      1688 2     [DSC$K_DTYPE_W]:
: 1563      1689 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' word integer'), 0);
: 1564      1690 2
: 1565      1691 2     [DSC$K_DTYPE_L]:
: 1566      1692 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' longword integer'), 0);
: 1567      1693 2
: 1568      1694 2     [DSC$K_DTYPE_Q]:
: 1569      1695 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' quadword integer'), 0);
: 1570      1696 2
: 1571      1697 2     [DSC$K_DTYPE_F]:
: 1572      1698 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' F_floating'), 0);
: 1573      1699 2
: 1574      1700 2     [DSC$K_DTYPE_D]:
: 1575      1701 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' D_floating'), 0);
: 1576      1702 2
: 1577      1703 2     [DSC$K_DTYPE_FC]:
: 1578      1704 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' F_floating complex'), 0);
: 1579      1705 2
: 1580      1706 2     [DSC$K_DTYPE_DC]:
: 1581      1707 2         DBG$PRINT(UPLIT BYTE(%ASCIC ' D_complex'), 0);
: 1582      1708 2
: 1583      1709 2     [DSC$K_DTYPE_I]:
: 1584      1710 2
```



|      |      |                                                                              |
|------|------|------------------------------------------------------------------------------|
| 1585 | 1711 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' character-coded text'), 0);                   |
| 1586 | 1712 |                                                                              |
| 1587 | 1713 | [DSC\$K_DTYPE_NU]:                                                           |
| 1588 | 1714 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' numeric string, unsigned'), 0);               |
| 1589 | 1715 |                                                                              |
| 1590 | 1716 | [DSC\$K_DTYPE_NL]:                                                           |
| 1591 | 1717 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' numeric string, left separate sign'), 0);     |
| 1592 | 1718 |                                                                              |
| 1593 | 1719 | [DSC\$K_DTYPE_NLO]:                                                          |
| 1594 | 1720 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' numeric string, left overpunched sign'), 0);  |
| 1595 | 1721 |                                                                              |
| 1596 | 1722 | [DSC\$K_DTYPE_NR]:                                                           |
| 1597 | 1723 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' numeric string, right separate sign'), 0);    |
| 1598 | 1724 |                                                                              |
| 1599 | 1725 | [DSC\$K_DTYPE_NRO]:                                                          |
| 1600 | 1726 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' numeric string, right overpunched sign'), 0); |
| 1601 | 1727 |                                                                              |
| 1602 | 1728 | [DSC\$K_DTYPE_NZ]:                                                           |
| 1603 | 1729 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' numeric string, zoned sign'), 0);             |
| 1604 | 1730 |                                                                              |
| 1605 | 1731 | [DSC\$K_DTYPE_P]:                                                            |
| 1606 | 1732 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' packed decimal string'), 0);                  |
| 1607 | 1733 |                                                                              |
| 1608 | 1734 | [DSC\$K_DTYPE_ZI]:                                                           |
| 1609 | 1735 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' sequence of instructions'), 0);               |
| 1610 | 1736 |                                                                              |
| 1611 | 1737 | [DSC\$K_DTYPE_ZEM]:                                                          |
| 1612 | 1738 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' procedure entry mask'), 0);                   |
| 1613 | 1739 |                                                                              |
| 1614 | 1740 | [DSC\$K_DTYPE_DSC]:                                                          |
| 1615 | 1741 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' descriptor'), 0);                             |
| 1616 | 1742 |                                                                              |
| 1617 | 1743 | [DSC\$K_DTYPE_OU]:                                                           |
| 1618 | 1744 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' octaword logical'), 0);                       |
| 1619 | 1745 |                                                                              |
| 1620 | 1746 | [DSC\$K_DTYPE_OI]:                                                           |
| 1621 | 1747 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' octaword integer'), 0);                       |
| 1622 | 1748 |                                                                              |
| 1623 | 1749 | [DSC\$K_DTYPE_G]:                                                            |
| 1624 | 1750 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' G_floating'), 0);                             |
| 1625 | 1751 |                                                                              |
| 1626 | 1752 | [DSC\$K_DTYPE_H]:                                                            |
| 1627 | 1753 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' H_floating'), 0);                             |
| 1628 | 1754 |                                                                              |
| 1629 | 1755 | [DSC\$K_DTYPE_GC]:                                                           |
| 1630 | 1756 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' G_floating complex'), 0);                     |
| 1631 | 1757 |                                                                              |
| 1632 | 1758 | [DSC\$K_DTYPE_HC]:                                                           |
| 1633 | 1759 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' H_floating complex'), 0);                     |
| 1634 | 1760 |                                                                              |
| 1635 | 1761 | [DSC\$K_DTYPE_CIT]:                                                          |
| 1636 | 1762 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' COBOL intermediate temporary'), 0);           |
| 1637 | 1763 |                                                                              |
| 1638 | 1764 | [DSC\$K_DTYPE_BPV]:                                                          |
| 1639 | 1765 | DBG\$PRINT(UPLIT BYTE(%ASCIC ' bound procedure value'), 0);                  |
| 1640 | 1766 |                                                                              |
| 1641 | 1767 | [DSC\$K_DTYPE_BLV]:                                                          |



```
: 1642      1768      2      DBG$PRINT(UPLIT BYTE(%ASCIC ' bound label value'), 0);
: 1643      1769
: 1644      1770      [DSC$K_DTYPE_VU]:
: 1645      1771      DBG$PRINT(UPLIT BYTE(%ASCIC ' bit unaligned'), 0);
: 1646      1772
: 1647      1773      [DSC$K_DTYPE_VT]:
: 1648      1774      DBG$PRINT(UPLIT BYTE(%ASCIC ' varying text'), 0);
: 1649      1775
: 1650      1776      [INRANGE, OUTRANGE]:
: 1651      1777      BEGIN
: 1652      1778
: 1653      1779      ! This is the only one which is not VAX Standard Data Type.
: 1654      1780      !
: 1655      1781      !
: 1656      1782      IF .TYPECODE EQL DST$K_BOOL
: 1657      1783      THEN
: 1658      1784      DBG$PRINT(UPLIT BYTE(%ASCIC ' boolean'), 0)
: 1659      1785      ELSE
: 1660      1786      DBG$PRINT(UPLIT BYTE(%ASCIC ' undefined type'), 0);
: 1661      1787
: 1662      1788      END;
: 1663      1789
: 1664      1790      TES;
: 1665      1791
: 1666      1792      RETURN 0;
: 1667      1793
: 1668      1794      END;
```

```
.PSECT DBG$PLIT,NOWRT, SHR, PIC,0

69 6C 61 63 64 6C 65 69 66 20 74 69 62 20 0A 0038A P.ABZ: .ASCII <10>\ bit field\
6C 61 63 69 67 6F 6C 20 65 74 79 62 20 0D 00395 P.ACA: .ASCII <13>\ byte logical\
67 6F 6C 69 67 6F 6C 20 64 72 6F 77 20 0D 003A3 P.ACB: .ASCII <13>\ word logical\
67 6F 6C 20 64 72 6F 77 67 6E 6F 6C 20 11 003B1 P.ACC: .ASCII <17>\ longword logical\
67 6F 6C 20 64 72 6F 77 64 61 75 71 20 11 003C3 P.ACD: .ASCII <17>\ quadword logical\
72 65 67 65 74 6E 69 20 65 74 79 62 20 0D 003D5 P.ACE: .ASCII <13>\ byte integer\
72 65 67 65 74 6E 69 20 64 72 6F 77 20 0D 003E3 P.ACF: .ASCII <13>\ word integer\
65 74 6E 69 20 64 72 6F 77 67 6E 6F 6C 20 11 003F1 P.ACG: .ASCII <17>\ longword integer\
65 74 6E 69 20 64 72 6F 77 64 61 75 71 20 11 00400
72 65 67 00400
65 74 6E 69 20 64 72 6F 77 64 61 75 71 20 11 00403 P.ACH: .ASCII <17>\ quadword integer\
67 6E 69 74 61 6F 6C 66 5F 46 20 0B 00415 P.ACI: .ASCII <11>\ F_floating\
67 6E 69 74 61 6F 6C 66 5F 44 20 0B 00421 P.ACJ: .ASCII <11>\ D_floating\
6F 63 20 67 6E 69 74 61 6F 6C 66 5F 46 20 13 0042D P.ACK: .ASCII <19>\ F_floating complex\
78 65 6C 70 6D 6F 63 5F 44 20 0A 0043C
64 6F 63 2D 72 65 74 63 61 72 61 68 63 20 15 00441 P.ACL: .ASCII <10>\ D_complex\
74 78 65 74 20 64 65 0044C P.ACM: .ASCII <21>\ character-coded text\
6E 69 72 74 73 20 63 69 72 65 6D 75 6E 20 19 0045B
6E 69 72 74 73 20 63 69 72 65 6D 75 6E 20 2C 67 00462 P.ACN: .ASCII <25>\ numeric string, unsigned\
74 61 72 61 70 65 73 20 74 66 65 6C 20 2C 67 00471
6E 69 72 74 73 20 63 69 72 65 6D 75 6E 20 23 0047C P.ACO: .ASCII \# numeric string, left separate sign\
74 61 72 61 70 65 73 20 74 66 65 6C 20 2C 67 0048B
6E 67 69 73 20 65 0049A
```



4S-1S,-  
5S-1S,-  
6S-1S,-  
7S-1S,-  
8S-1S,-  
9S-1S,-  
10S-1S,-  
11S-1S,-  
12S-1S,-  
13S-1S,-

: 1654  
:  
:  
: 1668



|          |    |      |    |    |       |       |            |                |      |
|----------|----|------|----|----|-------|-------|------------|----------------|------|
|          |    |      |    |    |       |       | 14\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 15\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 16\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 17\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 18\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 20\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 22\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 24\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 26\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 28\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 30\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 32\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 34\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 36\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 38\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 40\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 42\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 44\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 46\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 48\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 50\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 52\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 54\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 56\$-1\$,- |                |      |
|          |    |      |    |    |       |       | 2\$-1\$,-  |                |      |
|          |    |      |    |    |       |       | 2\$-1\$,-  |                |      |
|          |    |      |    |    |       |       | 58\$-1\$   |                |      |
| 0000009E | 8F | 04   | AC | D1 | 00058 | 2\$:  | CMPL       | TYPECODE, #158 | 1782 |
|          |    |      | 06 | 12 | 00060 |       | BNEQ       | 3\$            |      |
|          |    |      | 7E | D4 | 00062 |       | CLRL       | -(SP)          | 1784 |
|          |    |      | 52 | DD | 00064 |       | PUSHL      | R2             |      |
|          |    |      | 7D | 11 | 00066 |       | BRB        | 19\$           |      |
|          |    |      | 7E | D4 | 00068 | 3\$:  | CLRL       | -(SP)          | 1786 |
|          |    | 09   | A2 | 9F | 0006A |       | PUSHAB     | P.ADJ          |      |
|          |    |      | 7E | 11 | 0006D |       | BRB        | 21\$           |      |
|          |    |      | 7E | D4 | 0006F | 4\$:  | CLRL       | -(SP)          | 1672 |
|          |    | FD3E | C2 | 9F | 00071 |       | PUSHAB     | P.ABZ          |      |
|          |    |      | 7E | 11 | 00075 |       | BRB        | 23\$           |      |
|          |    |      | 7E | D4 | 00077 | 5\$:  | CLRL       | -(SP)          | 1675 |
|          |    | FD49 | C2 | 9F | 00079 |       | PUSHAB     | P.ACA          |      |
|          |    |      | 7E | 11 | 0007D |       | BRB        | 25\$           |      |
|          |    |      | 7E | D4 | 0007F | 6\$:  | CLRL       | -(SP)          | 1678 |
|          |    | FD57 | C2 | 9F | 00081 |       | PUSHAB     | P.ACB          |      |
|          |    |      | 7E | 11 | 00085 |       | BRB        | 27\$           |      |
|          |    |      | 7E | D4 | 00087 | 7\$:  | CLRL       | -(SP)          | 1681 |
|          |    | FD65 | C2 | 9F | 00089 |       | PUSHAB     | P.ACC          |      |
|          |    |      | 7E | 11 | 0008D |       | BRB        | 29\$           |      |
|          |    |      | 7E | D4 | 0008F | 8\$:  | CLRL       | -(SP)          | 1684 |
|          |    | FD77 | C2 | 9F | 00091 |       | PUSHAB     | P.ACD          |      |
|          |    |      | 7E | 11 | 00095 |       | BRB        | 31\$           |      |
|          |    |      | 7E | D4 | 00097 | 9\$:  | CLRL       | -(SP)          | 1687 |
|          |    | FD89 | C2 | 9F | 00099 |       | PUSHAB     | P.ACE          |      |
|          |    |      | 7E | 11 | 0009D |       | BRB        | 33\$           |      |
|          |    |      | 7E | D4 | 0009F | 10\$: | CLRL       | -(SP)          | 1690 |
|          |    | FD97 | C2 | 9F | 000A1 |       | PUSHAB     | P.ACF          |      |
|          |    |      | 7E | 11 | 000A5 |       | BRB        | 35\$           |      |
|          |    |      | 7E | D4 | 000A7 | 11\$: | CLRL       | -(SP)          | 1693 |



|      |    |    |       |       |        |       |      |
|------|----|----|-------|-------|--------|-------|------|
| FDA5 | C2 | 9F | 000A9 |       | PUSHAB | P.ACG |      |
|      | 7E | 11 | 000AD |       | BRB    | 37\$  |      |
|      | 7E | D4 | 000AF | 12\$: | CLRL   | -(SP) | 1696 |
| FDB7 | C2 | 9F | 000B1 |       | PUSHAB | P.ACH |      |
|      | 7E | 11 | 000B5 |       | BRB    | 39\$  |      |
|      | 7E | D4 | 000B7 | 13\$: | CLRL   | -(SP) | 1699 |
| FDC9 | C2 | 9F | 000B9 |       | PUSHAB | P.ACI |      |
|      | 7E | 11 | 000BD |       | BRB    | 41\$  |      |
|      | 7E | D4 | 000BF | 14\$: | CLRL   | -(SP) | 1702 |
| FDD5 | C2 | 9F | 000C1 |       | PUSHAB | P.ACJ |      |
|      | 7E | 11 | 000C5 |       | BRB    | 43\$  |      |
|      | 7E | D4 | 000C7 | 15\$: | CLRL   | -(SP) | 1705 |
| FDE1 | C2 | 9F | 000C9 |       | PUSHAB | P.ACK |      |
|      | 7E | 11 | 000CD |       | BRB    | 45\$  |      |
|      | 7E | D4 | 000CF | 16\$: | CLRL   | -(SP) | 1708 |
| FDF5 | C2 | 9F | 000D1 |       | PUSHAB | P.ACL |      |
|      | 7E | 11 | 000D5 |       | BRB    | 47\$  |      |
|      | 7E | D4 | 000D7 | 17\$: | CLRL   | -(SP) | 1711 |
| FE00 | C2 | 9F | 000D9 |       | PUSHAB | P.ACM |      |
|      | 7D | 11 | 000DD |       | BRB    | 49\$  |      |
|      | 7E | D4 | 000DF | 18\$: | CLRL   | -(SP) | 1714 |
| FE16 | C2 | 9F | 000E1 |       | PUSHAB | P.ACN |      |
|      | 7C | 11 | 000E5 | 19\$: | BRB    | 51\$  |      |
|      | 7E | D4 | 000E7 | 20\$: | CLRL   | -(SP) | 1717 |
| FE30 | C2 | 9F | 000E9 |       | PUSHAB | P.ACO |      |
|      | 7B | 11 | 000ED | 21\$: | BRB    | 53\$  |      |
|      | 7E | D4 | 000EF | 22\$: | CLRL   | -(SP) | 1720 |
| FE54 | C2 | 9F | 000F1 |       | PUSHAB | P.ACP |      |
|      | 7A | 11 | 000F5 | 23\$: | BRB    | 55\$  |      |
|      | 7E | D4 | 000F7 | 24\$: | CLRL   | -(SP) | 1723 |
| FE7B | C2 | 9F | 000F9 |       | PUSHAB | P.ACQ |      |
|      | 79 | 11 | 000FD | 25\$: | BRB    | 57\$  |      |
|      | 7E | D4 | 000FF | 26\$: | CLRL   | -(SP) | 1726 |
| FEA0 | C2 | 9F | 00101 |       | PUSHAB | P.ACR |      |
|      | 78 | 11 | 00105 | 27\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 00107 | 28\$: | CLRL   | -(SP) | 1729 |
| FEC8 | C2 | 9F | 00109 |       | PUSHAB | P.ACS |      |
|      | 70 | 11 | 0010D | 29\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 0010F | 30\$: | CLRL   | -(SP) | 1732 |
| FEE4 | C2 | 9F | 00111 |       | PUSHAB | P.ACT |      |
|      | 68 | 11 | 00115 | 31\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 00117 | 32\$: | CLRL   | -(SP) | 1735 |
| FEFB | C2 | 9F | 00119 |       | PUSHAB | P.ACU |      |
|      | 60 | 11 | 0011D | 33\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 0011F | 34\$: | CLRL   | -(SP) | 1738 |
| FF15 | C2 | 9F | 00121 |       | PUSHAB | P.ACV |      |
|      | 58 | 11 | 00125 | 35\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 00127 | 36\$: | CLRL   | -(SP) | 1741 |
| FF2B | C2 | 9F | 00129 |       | PUSHAB | P.ACW |      |
|      | 50 | 11 | 0012D | 37\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 0012F | 38\$: | CLRL   | -(SP) | 1744 |
| FF37 | C2 | 9F | 00131 |       | PUSHAB | P.ACX |      |
|      | 48 | 11 | 00135 | 39\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 00137 | 40\$: | CLRL   | -(SP) | 1747 |
| FF49 | C2 | 9F | 00139 |       | PUSHAB | P.ACY |      |
|      | 40 | 11 | 0013D | 41\$: | BRB    | 59\$  |      |
|      | 7E | D4 | 0013F | 42\$: | CLRL   | -(SP) | 1750 |



|      |    |    |       |       |        |                |      |
|------|----|----|-------|-------|--------|----------------|------|
| FF5B | C2 | 9F | 00141 |       | PUSHAB | P.ACZ          |      |
|      | 38 | 11 | 00145 | 43\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 00147 | 44\$: | CLRL   | -(SP)          | 1753 |
| FF67 | C2 | 9F | 00149 |       | PUSHAB | P.ADA          |      |
|      | 30 | 11 | 0014D | 45\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 0014F | 46\$: | CLRL   | -(SP)          | 1756 |
| FF73 | C2 | 9F | 00151 |       | PUSHAB | P.ADB          |      |
|      | 28 | 11 | 00155 | 47\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 00157 | 48\$: | CLRL   | -(SP)          | 1759 |
| 87   | A2 | 9F | 00159 |       | PUSHAB | P.ADC          |      |
|      | 21 | 11 | 0015C | 49\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 0015E | 50\$: | CLRL   | -(SP)          | 1762 |
| 9B   | A2 | 9F | 00160 |       | PUSHAB | P.ADD          |      |
|      | 1A | 11 | 00163 | 51\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 00165 | 52\$: | CLRL   | -(SP)          | 1765 |
| B9   | A2 | 9F | 00167 |       | PUSHAB | P.ADE          |      |
|      | 13 | 11 | 0016A | 53\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 0016C | 54\$: | CLRL   | -(SP)          | 1768 |
| D0   | A2 | 9F | 0016E |       | PUSHAB | P.ADF          |      |
|      | 0C | 11 | 00171 | 55\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 00173 | 56\$: | CLRL   | -(SP)          | 1771 |
| E3   | A2 | 9F | 00175 |       | PUSHAB | P.ADG          |      |
|      | 05 | 11 | 00178 | 57\$: | BRB    | 59\$           |      |
|      | 7E | D4 | 0017A | 58\$: | CLRL   | -(SP)          | 1774 |
| F2   | A2 | 9F | 0017C |       | PUSHAB | P.ADH          |      |
|      | 02 | FB | 0017F | 59\$: | CALLS  | #2, DBG\$PRINT |      |
|      |    | 04 | 00186 |       | RET    |                | 1794 |

00000000G 00

; Routine Size: 391 bytes, Routine Base: DBG\$CODE + 0936



```

: 1670      1795 1 GLOBAL ROUTINE DBG$TRANSLATE_REG(REG_NUM): NOVALUE =
: 1671      1796 1
: 1672      1797 1 FUNCTION
: 1673      1798 1     This routine translate register number into register name.
: 1674      1799 1
: 1675      1800 1 INPUTS
: 1676      1801 1     REG_NUM - register number (from 0 to 15).
: 1677      1802 1
: 1678      1803 1 OUTPUTS
: 1679      1804 1     None.
: 1680      1805 1
: 1681      1806 1
: 1682      1807 2 BEGIN
: 1683      1808 2
: 1684      1809 2 IF (.REG_NUM GEQ 0) AND (.REG_NUM LEQ 11)
: 1685      1810 2 THEN
: 1686      1811 2     DBG$PRINT(UPLIT BYTE(%ASCIC '%R!UL'), .REG_NUM)
: 1687      1812 2 ELSE
: 1688      1813 2     BEGIN
: 1689      1814 2     CASE .REG_NUM FROM 12 TO 15 OF
: 1690      1815 2     SET
: 1691      1816 2     [12]: DBG$PRINT(UPLIT BYTE(%ASCIC '%AP'), 0);
: 1692      1817 2     [13]: DBG$PRINT(UPLIT BYTE(%ASCIC '%FP'), 0);
: 1693      1818 2     [14]: DBG$PRINT(UPLIT BYTE(%ASCIC '%SP'), 0);
: 1694      1819 2     [15]: DBG$PRINT(UPLIT BYTE(%ASCIC '%PC'), 0);
: 1695      1820 2     TES;
: 1696      1821 2     END;
: 1697      1822 2
: 1698      1823 2 RETURN 0;
: 1699      1824 1 END;

```

.PSECT DBG\$PLIT,NOWRT, SHR, PIC,0

```

4C 55 21 52 25 05 00665 P.ADK: .ASCII <5>\%R!UL\
50 41 25 03 0066B P.ADL: .ASCII <3>\%AP\
50 46 25 03 0066F P.ADM: .ASCII <3>\%FP\
50 53 25 03 00673 P.ADN: .ASCII <3>\%SP\
43 50 25 03 00677 P.ADO: .ASCII <3>\%PC\

```

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

|      |      |           |      |            |        |                             |        |
|------|------|-----------|------|------------|--------|-----------------------------|--------|
|      |      |           |      | 0004 00000 | .ENTRY | DBG\$TRANSLATE_REG, Save R2 | : 1795 |
|      | 52   | 00000000' | EF   | 9E 00002   | MOVAB  | P.ADK, R2                   | : 1809 |
|      | 50   | 04        | AC   | D0 00009   | MOVL   | REG_NUM, R0                 | :      |
|      |      |           | 0B   | 19 0000D   | BLSS   | 1\$                         | :      |
|      | 0B   |           | 50   | D1 0000F   | CMPL   | R0, #11                     | :      |
|      |      |           | 06   | 14 00012   | BGTR   | 1\$                         | :      |
|      |      |           | 50   | DD 00014   | PUSHL  | R0                          | : 1811 |
|      |      |           | 52   | DD 00016   | PUSHL  | R2                          | :      |
|      |      |           | 26   | 11 00018   | BRB    | 7\$                         | :      |
|      |      |           | 50   | CF 0001A   | CASEL  | R0, #12, #3                 | : 1814 |
| 001D | 03   | 0C        |      |            | .WORD  | 3\$-2\$,-                   | :      |
|      | 0016 | 000F      | 0008 | 0001E      |        | 4\$-2\$,-                   | :      |



|    |    |    |       |      |           |                |
|----|----|----|-------|------|-----------|----------------|
|    |    |    |       |      | 5\$-2\$,- |                |
|    |    |    |       |      | 6\$-2\$   |                |
| 06 | 7E | D4 | 00026 | 3\$: | CLRL      | -(SP)          |
|    | A2 | 9F | 00028 |      | PUSHAB    | P.ADL          |
|    | 13 | 11 | 0002B |      | BRB       | 7\$            |
|    | 7E | D4 | 0002D | 4\$: | CLRL      | -(SP)          |
| 0A | A2 | 9F | 0002F |      | PUSHAB    | P.ADM          |
|    | 0C | 11 | 00032 |      | BRB       | 7\$            |
|    | 7E | D4 | 00034 | 5\$: | CLRL      | -(SP)          |
| 0E | A2 | 9F | 00036 |      | PUSHAB    | P.ADN          |
|    | 05 | 11 | 00039 |      | BRB       | 7\$            |
|    | 7E | D4 | 0003B | 6\$: | CLRL      | -(SP)          |
| 12 | A2 | 9F | 0003D |      | PUSHAB    | P.ADO          |
|    | 02 | FB | 00040 | 7\$: | CALLS     | #2, DBG\$PRINT |
|    |    | 04 | 00047 |      | RET       |                |

00000000G 00

; Routine Size: 72 bytes, Routine Base: DBG\$CODE + 0ABD



```
: 1701 1825 1 GLOBAL ROUTINE DBGSDUMP_HEX(ADDRESS,LENGTH,RELADDR,HEADER_STRING): NOVALUE =
: 1702 1826 1
: 1703 1827 1 FUNCTION
: 1704 1828 1     This routine is taken from TOOL$:DBGDMPDST.B32, and is modified some.
: 1705 1829 1
: 1706 1830 1     This routine dumps the specified number of bytes, starting at the
: 1707 1831 1     specified address, in hex. The output is formatted as four longwords
: 1708 1832 1     to a line followed by the start address of the four longword unit. A
: 1709 1833 1     header, if supplied, is output at the begining of the first line of text.
: 1710 1834 1     The output text is right justified to 78 columns.
: 1711 1835 1
: 1712 1836 1 INPUTS
: 1713 1837 1     ADDRESS      - The start address of the block to be dumped.
: 1714 1838 1
: 1715 1839 1     LENGTH        - The size of the block to be dumped in bytes.
: 1716 1840 1
: 1717 1841 1     RELADDR       - The start address relative to the DST. This is the value
: 1718 1842 1                   printed at the end of an output line of text, as
: 1719 1843 1                   mentioned above.
: 1720 1844 1
: 1721 1845 1     HEADER_STRING - Start address of the header name in counted ASCII. This
: 1722 1846 1                   string is printed at the begining of the first line
: 1723 1847 1                   of text.
: 1724 1848 1
: 1725 1849 1 OUTPUTS
: 1726 1850 1     None
: 1727 1851 1
: 1728 1852 1
: 1729 1853 1 BEGIN
: 1730 1854 2
: 1731 1855 2 MAP
: 1732 1856 2     HEADER_STRING : REF VECTOR[,BYTE]; ! Location of ASCII header name.
: 1733 1857 2
: 1734 1858 2 LOCAL
: 1735 1859 2     BYTE_LENGTH,      ! Remaining size of input block in bytes.
: 1736 1860 2     OUTBLOCK : VECTOR[79,BYTE], ! Output buffer for storing the ASCII
: 1737 1861 2                                     hex characters.
: 1738 1862 2     OUTBLOCK_INDEX,   ! Variable to index OUTBLOCK.
: 1739 1863 2     REL_ADDRESS,     ! Address relative to start of DST.
: 1740 1864 2     START_ADDRESS, ! Start address of block to be dumped.
: 1741 1865 2     VAL_SIZE,       ! Size in bytes of the value to be
: 1742 1866 2                                     converted to ASCII hex.
: 1743 1867 2     STRNG_DESC : BLOCK[8,BYTE]; ! String descriptor passed as argument
: 1744 1868 2                                     to external routines.
: 1745 1869 2
: 1746 1870 2
: 1747 1871 2
: 1748 1872 2
: 1749 1873 2     ! The locally used variables are equated to the formal parameters.
: 1750 1874 2
: 1751 1875 2     START_ADDRESS = .ADDRESS;
: 1752 1876 2     BYTE_LENGTH  = .LENGTH;
: 1753 1877 2     REL_ADDRESS  = .RELADDR;
: 1754 1878 2
: 1755 1879 2
: 1756 1880 2     ! The index into OUTBLOCK is initialized for the fill-in.
: 1757 1881 2
```



```
: 1758      1882      2      OUTBLOCK_INDEX = 60;
: 1759      1883      2      VAL_SIZE = 4;
: 1760      1884      2
: 1761      1885      2
: 1762      1886      2      ! The fixed parts of the string descriptor are set up.
: 1763      1887      2
: 1764      1888      2      STRNG_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
: 1765      1889      2      STRNG_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
: 1766      1890      2      STRNG_DESC [DSC$W_LENGTH] = 8;
: 1767      1891      2
: 1768      1892      2
: 1769      1893      2      ! The output buffer is cleared & the header string shifted into
: 1770      1894      2      ! the first part of the buffer.
: 1771      1895      2
: 1772      1896      2      CH$FILL(' ',79,OUTBLOCK[0]);
: 1773      1897      2      CH$MOVE(.HEADER_STRING[0],HEADER_STRING[1],OUTBLOCK[1]);
: 1774      1898      2
: 1775      1899      2
: 1776      1900      2      ! The input buffer is looped through, converting one longword (or trailing
: 1777      1901      2      ! bytes) at a time to hex, till byte count is exhausted.
: 1778      1902      2
: 1779      1903      2      WHILE .BYTE_LENGTH GTR 0 DO
: 1780      1904      2          BEGIN
: 1781      1905      2
: 1782      1906      2          ! The minimum number of digits to dump is adjusted if there is less
: 1783      1907      2          ! than a longword remaining. The conversion to hex is made & the buffer
: 1784      1908      2          ! pointers updated.
: 1785      1909      2
: 1786      1910      2          IF .BYTE_LENGTH LSS 4 THEN VAL_SIZE = .BYTE_LENGTH;
: 1787      1911      2          STRNG_DESC [DSC$A_POINTER] = OUTBLOCK[OUTBLOCK_INDEX];
: 1788      1912      2          OT$SCVT_L_TZ(.START_ADDRESS,STRNG_DESC,(2*.VAL_SIZE),.VAL_SIZE);
: 1789      1913      2          BYTE_LENGTH = .BYTE_LENGTH - 4;
: 1790      1914      2          START_ADDRESS = .START_ADDRESS + 4;
: 1791      1915      2          OUTBLOCK_INDEX = .OUTBLOCK_INDEX - 10;
: 1792      1916      2
: 1793      1917      2
: 1794      1918      2
: 1795      1919      2      ! If the buffer is full or the byte count is exhausted, a text line
: 1796      1920      2      ! is output.
: 1797      1921      2
: 1798      1922      4      IF (.OUTBLOCK_INDEX LSS 30) OR (.BYTE_LENGTH LEQ 0)
: 1799      1923      3      THEN
: 1800      1924      4          BEGIN
: 1801      1925      4
: 1802      1926      4
: 1803      1927      4          ! The string descriptor is set up to receive the converted address.
: 1804      1928      4          ! The rel address is converted, stored, updated for the next pass &
: 1805      1929      4          ! the colon separating the address from text is entered.
: 1806      1930      4
: 1807      1931      4          IF .REL_ADDRESS GEQ %X'10000'
: 1808      1932      4          THEN
: 1809      1933      4              STRNG_DESC [DSC$W_LENGTH] = 8
: 1810      1934      4
: 1811      1935      4          ELSE
: 1812      1936      4              STRNG_DESC [DSC$W_LENGTH] = 4;
: 1813      1937      4
: 1814      1938      4          STRNG_DESC [DSC$A_POINTER] = OUTBLOCK[71];
```



```
: 1815      1939  4      OTSS$CVT L TZ(REL_ADDRESS,STRNG_DESC,,STRNG_DESC[DSC$W_LENGTH],4);
: 1816      1940  4      REL_ADDRESS = .REL_ADDRESS + 16;
: 1817      1941  4      OUTBLOCK[70] = ':';
: 1818      1942  4
: 1819      1943  4
: 1820      1944  4      ! The filled buffer is output as a line of text.
: 1821      1945  4      !
: 1822      1946  4      STRNG_DESC [DSC$W_LENGTH] = 78;
: 1823      1947  4      STRNG_DESC [DSC$A_POINTER] = OUTBLOCK[1];
: 1824      1948  4      OUTBLOCK[0] = 78;
: 1825      1949  4      DBG$PRINT(UPLIT BYTE(%ASCIC '!AC'), OUTBLOCK);
: 1826      1950  4      DBG$NEWLINE();
: 1827      1951  4
: 1828      1952  4
: 1829      1953  4      ! OUTBLOCK is cleared for more input and loop parameters reset.
: 1830      1954  4      !
: 1831      1955  4      STRNG_DESC [DSC$W_LENGTH] = 8;
: 1832      1956  4      CH$FILL(' ',79,OUTBLOCK[0]);
: 1833      1957  4      OUTBLOCK_INDEX = 60;
: 1834      1958  3      END;
: 1835      1959  3
: 1836      1960  2      END;
: 1837      1961  2
: 1838      1962  2
: 1839      1963  2      ! After all bytes have been successfully dumped, control is returned to
: 1840      1964  2      ! the calling routine.
: 1841      1965  2      !
: 1842      1966  2      RETURN 0;
: 1843      1967  1      END;
```

.PSECT DBG\$PLIT,NOWRT, SHR, PIC,0

43 41 21 03 0067B P.ADP: .ASCII <3>\!AC\ ;

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

```
.ENTRY DBG$DUMP_HEX, Save R2,R3,R4,R5,R6,R7,R8,R9,-: 1825
R10
MOVAB OTSS$CVT_L TZ, R10
MOVAB -88(SP), SP
MOVL ADDRESS, START_ADDRESS 1875
MOVL LENGTH, BYTE_LENGTH 1876
PUSHL RELADDR 1877
MOVL #60, OUTBLOCK_INDEX 1882
MOVL #4, VAL_SIZE 1883
MOVL #17694728, STRNG_DESC 1890
MOVC5 #0, (SP), #32, #79, OUTBLOCK 1896

MOVL HEADER_STRING, R0 1897
MOVZBL (R0), R1
MOVC3 R1, 1(R0), OUTBLOCK+1
TSTL BYTE_LENGTH 1903
BGTR 2$
```

```
004F 8F 20 04 5A 00000000G 00 9E 00002
5E A8 AE 9E 00009
59 04 AC D0 0000D
57 08 AC D0 00011
OC AC DD 00015
56 3C D0 00018
58 04 D0 0001B
AE 010E0008 8F D0 0001E
6E 00 2C 00026
OC AE 0002D
50 10 AC D0 0002F
51 60 9A 00033
OD AE 01 A0 51 28 00036 1$:
57 D5 0003C
01 14 0003E
```



|      |          |           |           |      |             |      |        |                                        |  |      |
|------|----------|-----------|-----------|------|-------------|------|--------|----------------------------------------|--|------|
|      |          |           |           | 04   | 00040       |      | RET    |                                        |  |      |
|      |          |           |           | 57   | D1 00041    | 2\$: | CMPL   | BYTE_LENGTH, #4                        |  | 1911 |
|      |          |           |           | 03   | 18 00044    |      | BGEQ   | 3\$                                    |  |      |
|      |          |           |           | 57   | D0 00046    |      | MOVL   | BYTE_LENGTH, VAL_SIZE                  |  |      |
|      | 08       | AE        | 0C        | AE46 | 9E 00049    | 3\$: | MOVAB  | OUTBLOCK[OUTBLOCK_INDEX], STRNG_DESC+4 |  | 1912 |
|      |          |           |           | 58   | DD 0004F    |      | PUSHL  | VAL_SIZE                               |  | 1913 |
| 7E   |          | 58        |           | 01   | 78 00051    |      | ASHL   | #1, VAL_SIZE, -(SP)                    |  |      |
|      |          |           | 0C        | AE   | 9F 00055    |      | PUSHAB | STRNG_DESC                             |  |      |
|      |          |           |           | 59   | DD 00058    |      | PUSHL  | START_ADDRESS                          |  |      |
|      |          | 6A        |           | 04   | FB 0005A    |      | CALLS  | #4, OTSSCVT L TZ                       |  |      |
|      |          | 57        |           | 04   | C2 0005D    |      | SUBL2  | #4, BYTE_LENGTH                        |  | 1914 |
|      |          | 59        |           | 04   | C0 00060    |      | ADDL2  | #4, START_ADDRESS                      |  | 1915 |
|      |          | 56        |           | 0A   | C2 00063    |      | SUBL2  | #10, OUTBLOCK_INDEX                    |  | 1916 |
|      |          | 1E        |           | 56   | D1 00066    |      | CMPL   | OUTBLOCK_INDEX, #30                    |  | 1922 |
|      |          |           |           | 04   | 19 00069    |      | BLSS   | 4\$                                    |  |      |
|      |          |           |           | 57   | D5 0006B    |      | TSTL   | BYTE_LENGTH                            |  |      |
|      |          |           |           | CD   | 14 0006D    |      | BGTR   | 1\$                                    |  |      |
|      | 00010000 | 8F        |           | 6E   | D1 0006F    | 4\$: | CMPL   | REL_ADDRESS, #65536                    |  | 1931 |
|      |          |           |           | 05   | 19 00076    |      | BLSS   | 5\$                                    |  |      |
|      |          | 04        | AE        | 08   | B0 00078    |      | MOVW   | #8, STRNG_DESC                         |  | 1933 |
|      |          |           |           | 04   | 11 0007C    |      | BRB    | 6\$                                    |  |      |
|      |          | 04        | AE        | 04   | B0 0007E    | 5\$: | MOVW   | #4, STRNG_DESC                         |  | 1936 |
|      |          | 08        | AE        | 53   | AE 9E 00082 | 6\$: | MOVAB  | OUTBLOCK+71, STRNG_DESC+4              |  | 1938 |
|      |          |           |           | 04   | DD 00087    |      | PUSHL  | #4                                     |  | 1939 |
|      |          | 7E        |           | 08   | AE 3C 00089 |      | MOVZWL | STRNG_DESC, -(SP)                      |  |      |
|      |          |           | 0C        | AE   | 9F 0008D    |      | PUSHAB | STRNG_DESC                             |  |      |
|      |          |           | 0C        | AE   | 9F 00090    |      | PUSHAB | REL_ADDRESS                            |  |      |
|      |          |           |           | 04   | FB 00093    |      | CALLS  | #4, OTSSCVT L TZ                       |  |      |
|      |          | 6A        |           | 10   | C0 00096    |      | ADDL2  | #16, REL_ADDRESS                       |  | 1940 |
|      |          | 52        | AE        | 3A   | 90 00099    |      | MOVB   | #58, OUTBLOCK+70                       |  | 1941 |
|      |          | 04        | AE        | 4E   | 8F 9B 0009D |      | MOVZBW | #78, STRNG_DESC                        |  | 1946 |
|      |          | 08        | AE        | 0D   | AE 9E 000A2 |      | MOVAB  | OUTBLOCK+1, STRNG_DESC+4               |  | 1947 |
|      |          | 0C        | AE        | 4E   | 8F 90 000A7 |      | MOVB   | #78, OUTBLOCK                          |  | 1948 |
|      |          |           |           | 0C   | AE 9F 000AC |      | PUSHAB | OUTBLOCK                               |  | 1949 |
|      |          |           | 00000000' | EF   | 9F 000AF    |      | PUSHAB | P.ADP                                  |  |      |
|      |          | 00000000G | 00        | 02   | FB 000B5    |      | CALLS  | #2, DBG\$PRINT                         |  |      |
|      |          | 00000000G | 00        | 00   | FB 000BC    |      | CALLS  | #0, DBG\$NEWLINE                       |  | 1950 |
|      |          |           | 04        | AE   | 08 B0 000C3 |      | MOVW   | #8, STRNG_DESC                         |  | 1955 |
| 004F | 8F       | 20        |           | 00   | 2C 000C7    |      | MOVC5  | #0, (SP), #32, #79, OUTBLOCK           |  | 1956 |
|      |          |           |           | 0C   | AE 000CE    |      |        |                                        |  |      |
|      |          |           |           | 56   | 3C D0 000D0 |      | MOVL   | #60, OUTBLOCK_INDEX                    |  | 1957 |
|      |          |           |           | FF66 | 31 000D3    |      | BRW    | 1\$                                    |  | 1903 |
|      |          |           |           | 04   | 000D6       |      | RET    |                                        |  | 1967 |

; Routine Size: 215 bytes, Routine Base: DBG\$CODE + 0B05

; 1844 1968 1



```

: 1846 1969 1 ROUTINE BLD_SCOPE_SEARCH_LIST(NCANDS, SCOPE): NOVALUE =
: 1847 1970 1
: 1848 1971 1 FUNCTION
: 1849 1972 1 This routine builds a list of scopes for the candidate symbols
: 1850 1973 1 to be searched. If this list has not been built, this routine
: 1851 1974 1 will create a memory block for this list. If this list is too
: 1852 1975 1 small, this routine will expand the list by getting more memory
: 1853 1976 1 blocks. Then the search scope is entered to the list. Each time
: 1854 1977 1 this routine is called, the scope count is up by 1.
: 1855 1978 1
: 1856 1979 1 INPUTS
: 1857 1980 1 NCANDS - Keep track the number of scopes on the scope search list.
: 1858 1981 1 SCOPE - The desired search scope for the candidate symbol.
: 1859 1982 1
: 1860 1983 1 OUTPUTS
: 1861 1984 1 None.
: 1862 1985 1
: 1863 1986 1
: 1864 1987 2 BEGIN
: 1865 1988 2
: 1866 1989 2 MAP
: 1867 1990 2 NCANDS: REF VECTOR[1]; ! Keep track of the number of scopes
: 1868 1991 2
: 1869 1992 2 LOCAL
: 1870 1993 2 OLDLIST: REF VECTOR[,LONG], ! Pointer to scope search list about
: 1871 1994 2 ! to be copied to a larger
: 1872 1995 2 ! memory block
: 1873 1996 2 SCOPE_CNT; ! Scope count
: 1874 1997 2
: 1875 1998 2
: 1876 1999 2 ! Update the scope count.
: 1877 2000 2
: 1878 2001 2 SCOPE_CNT = .NCANDS[0];
: 1879 2002 2 SCOPE_CNT = .SCOPE_CNT + 1;
: 1880 2003 2
: 1881 2004 2
: 1882 2005 2 ! If we do not yet have a scope search list memory block, get one and
: 1883 2006 2 initialize its first element to give the list size that will fit in
: 1884 2007 2 the block.
: 1885 2008 2
: 1886 2009 2 IF .DBG$_SCOPELIST EQL 0
: 1887 2010 2 THEN
: 1888 2011 2 BEGIN
: 1889 2012 2 DBG$_SCOPELIST = DBG$GET_MEMORY(11);
: 1890 2013 2 DBG$_SCOPELIST[0] = 10;
: 1891 2014 2 END;
: 1892 2015 2
: 1893 2016 2
: 1894 2017 2 ! Note that we have to expand the scope search list memory block if it
: 1895 2018 2 is too small.
: 1896 2019 2
: 1897 2020 2 IF .SCOPE_CNT GTR .DBG$_SCOPELIST[0]
: 1898 2021 2 THEN
: 1899 2022 2 BEGIN
: 1900 2023 2 OLDLIST = .DBG$_SCOPELIST;
: 1901 2024 2 DBG$_SCOPELIST = DBG$GET_MEMORY(.OLDLIST[0] + 11);
: 1902 2025 2 CH$MOVE(4*(.OLDLIST[0]+1), .OLDLIST, .DBG$_SCOPELIST);

```



DBG\$SYMBOL  
V04-000

M 16  
16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

VAX-11 Bliss-32 V4.0-742  
[DEBUG.SRC]DBG\$SYMBOL.B32;1

Page 63  
(11)

: 1903 2026 3  
: 1904 2027  
: 1905 2028  
: 1906 2029  
: 1907 2030  
: 1908 2031  
: 1909 2032  
: 1910 2033  
: 1911 2034  
: 1912 2035  
: 1913 2036  
: 1914 2037 1

DBG\$ SCOPELST[0] = .DBG\$ SCOPELST[0] + 10;  
DBG\$REL\_MEMORY(.OLDLST);  
END;

! Now enter the scope to this list, and set the count.

DBG\$ SCOPELST[.SCOPE\_CNT] = .SCOPE;  
NCANDS[0] = .SCOPE\_CNT;  
RETURN 0;

END;

03FC 0000 BLD\_SCOPE\_SEARCH\_LIST:

|              |           |    |       |       |        |                                   |        |
|--------------|-----------|----|-------|-------|--------|-----------------------------------|--------|
| 59           | 00000000G | 00 | 9E    | 00002 | WORD   | Save R2,R3,R4,R5,R6,R7,R8,R9      | : 1969 |
| 58           | 00000000' | EF | 9E    | 00009 | MOVAB  | DBG\$GET_MEMORY, R9               |        |
| 56           | 04        | BC | D0    | 00010 | MOVAB  | DBG\$ SCOPELST, R8                |        |
|              |           | 56 | D6    | 00014 | MOVL   | @NCANDS, SCOPE_CNT                | : 2001 |
|              |           | 68 | D5    | 00016 | INCL   | SCOPE_CNT                         | : 2002 |
|              |           | 0C | 12    | 00018 | TSTL   | DBG\$ SCOPELST                    | : 2009 |
|              |           | 0B | DD    | 0001A | BNEQ   | 1\$                               |        |
|              | 69        | 01 | FB    | 0001C | PUSHL  | #11                               | : 2012 |
|              |           | 50 | D0    | 0001F | CALLS  | #1, DBG\$GET_MEMORY               |        |
| 00           |           | 0A | D0    | 00022 | MOVL   | R0, DBG\$ SCOPELST                |        |
| 00           |           | 56 | D1    | 00026 | MOVL   | #10, @DBG\$ SCOPELST              | : 2013 |
|              |           | 27 | 15    | 0002A | CMPL   | SCOPE_CNT, @DBG\$ SCOPELST        | : 2020 |
|              |           | 68 | D0    | 0002C | BLEQ   | 2\$                               |        |
| 57           |           | 67 | D0    | 0002F | MOVL   | DBG\$ SCOPELST, OLDLST            | : 2023 |
| 52           |           | 0B | A2    | 9F    | MOVL   | (OLDLST), R2                      | : 2024 |
|              |           | 01 | FB    | 00035 | PUSHAB | 11(R2)                            |        |
| 69           |           | 50 | D0    | 00038 | CALLS  | #1, DBG\$GET_MEMORY               |        |
| 68           |           | 04 | C4    | 0003B | MOVL   | R0, DBG\$ SCOPELST                |        |
| 52           |           | 04 | C0    | 0003E | MULL2  | #4, R2                            | : 2025 |
| 52           |           | 52 | 28    | 00041 | ADDL2  | #4, R2                            |        |
| 00 B8        |           | 0A | C0    | 00046 | MOVC3  | R2, (OLDLST), @DBG\$ SCOPELST     |        |
|              |           | 57 | DD    | 0004A | ADDL2  | #10, @DBG\$ SCOPELST              | : 2026 |
| 00000000G 00 |           | 01 | FB    | 0004C | PUSHL  | OLDLST                            | : 2027 |
| 00 B846      |           | 0B | AC    | D0    | CALLS  | #1, DBG\$REL_MEMORY               |        |
| 04 BC        |           | 56 | D0    | 00059 | MOVL   | SCOPE, @DBG\$ SCOPELST[SCOPE_CNT] | : 2033 |
|              |           | 04 | 0005D | RET   | MOVL   | SCOPE_CNT, @NCANDS                | : 2034 |
|              |           |    |       |       |        |                                   | : 2037 |

; Routine Size: 94 bytes, Routine Base: DBG\$CODE + 0BDC



```
1916 2038 1 ROUTINE GET_BLISS_SPECIAL_CASES(DSTPTR): NOVALUE =
1917 2039 1
1918 2040 1 FUNCTION
1919 2041 1     This routine handles the data type information for BLISS SPECIAL CASES
1920 2042 1     DST Record. Its DST type is DST$K_BLI, RST kind is RST$K_DATA, and
1921 2043 1     RST Fcode is RST$K_TYPE_BLI_DATA.
1922 2044 1
1923 2045 1     The text printed will be of the form:
1924 2046 1
1925 2047 1     +--
1926 2048 1     ref  vector [UNITS,BWL[,signed]]  --+ ,[formal]
1927 2049 1           bitvector [UNITS]
1928 2050 1           block [UNITS,BWL]
1929 2051 1           blockvector [N,BS,BWL]
1930 2052 1     +--
1931 2053 1
1932 2054 1     UNITS -- the number of units.
1933 2055 1     BWL   -- BYTE, WORD or LONG.
1934 2056 1     N, BS -- the size values for BLOCKVECTOR as defined in the
1935 2057 1             BLISS-32 Language manual.
1936 2058 1
1937 2059 1 INPUT
1938 2060 1     DSTPTR - Pointer to DST$RECORD, fields can be (DST$BLI_FIELDS,
1939 2061 1             DST$BLI_VEC_FIELDS, DST$BLI_BITVEC_FIELDS,
1940 2062 1             DST$BLI_BLOCK_FIELDS, DST$BLI_BLKVEC_FIELDS).
1941 2063 1
1942 2064 1 OUTPUT
1943 2065 1     None.
1944 2066 1
1945 2067 1
1946 2068 2 BEGIN
1947 2069 2
1948 2070 2 MAP
1949 2071 2     DSTPTR: REF DST$RECORD;
1950 2072 2
1951 2073 2 LOCAL
1952 2074 2     TRAIL1: REF DST$BLI_TRAILER1, ! Pointer to trailer 1
1953 2075 2     TRAIL2: REF DST$BLI_TRAILER2, ! Pointer to trailer 2
1954 2076 2
1955 2077 2
1956 2078 2     ! Set up pointers to the trailing fields.
1957 2079 2
1958 2080 2     TRAIL1 = DSTPTR[DST$A_BLI_TRLR1] + .DSTPTR[DST$B_BLI_LNG];
1959 2081 2     TRAIL2 = TRAIL1[DST$A_BLI_TRLR2] + .TRAIL1[DST$B_BLI_NAME];
1960 2082 2     IF .DSTPTR[DST$V_BLI_REF]
1961 2083 2     THEN
1962 2084 2         DBG$PRINT(UPLIT BYTE(%ASCIC', ref'), 0)
1963 2085 2     ELSE
1964 2086 2         DBG$PRINT(UPLIT BYTE(%ASCIC','), 0);
1965 2087 2
1966 2088 2
1967 2089 2     ! Get the structure information of this symbol.
1968 2090 2
1969 2091 2     CASE .DSTPTR[DST$V_BLI_STRUC] FROM DST$K_BLI_NOSTRUC TO DST$K_BLI_BLKVEC OF
1970 2092 2         SET
1971 2093 2
1972 2094 2
```



```
1973 2095 2
1974 2096
1975 2097
1976 2098
1977 2099
1978 2100
1979 2101
1980 2102
1981 2103
1982 2104
1983 2105
1984 2106
1985 2107
1986 2108
1987 2109
1988 2110
1989 2111
1990 2112
1991 2113
1992 2114
1993 2115
1994 2116
1995 2117
1996 2118
1997 2119
1998 2120
1999 2121
2000 2122
2001 2123
2002 2124
2003 2125
2004 2126
2005 2127
2006 2128
2007 2129
2008 2130
2009 2131
2010 2132
2011 2133
2012 2134
2013 2135
2014 2136
2015 2137
2016 2138
2017 2139
2018 2140
2019 2141
2020 2142
2021 2143
2022 2144
2023 2145
2024 2146
2025 2147
2026 2148
2027 2149
2028 2150
2029 2151 2

! No structure information.
[DST$K_BLI_NOSTRUC]:
  BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC ' no field structure'), 0);
  END;

! Vector.
[DST$K_BLI_VEC]:
  BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC' vector[!UL,']), .DSTPTR[DST$L_BLI_VEC_UNITS]);
    BLIUNIT(DSTPTR[DST$V_BLI_VEC_UNIT_SIZE]);
    IF .DSTPTR[DST$V_BLI_VEC_SIGN_EXT]
    THEN
      DBG$PRINT(UPLIT BYTE(%ASCIC',signed']), 0)
    ELSE
      DBG$PRINT(UPLIT BYTE(%ASCIC']), 0);
  END;

! Bitvector.
[DST$K_BLI_BITVEC]:
  BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC ' bitvector[!UL]'), .DSTPTR[DST$L_BLI_BITVEC_SIZE]);
  END;

! Block.
[DST$K_BLI_BLOCK]:
  BEGIN
    DBG$PRINT(UPLIT BYTE
      (%ASCIC ' block[!UL,']), .DSTPTR[DST$L_BLI_BLOCK_UNITS]);
    BLIUNIT(DSTPTR[DST$V_BLI_BLOCK_UNIT_SIZE]);
    DBG$PRINT(UPLIT BYTE(%ASCIC '']), 0);
  END;

! Blockvector.
[DST$K_BLI_BLKVEC]:
  BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC ' blockvector[!UL,!UL,']),
      .DSTPTR[DST$L_BLI_BLKVEC_BLOCKS],
      .DSTPTR[DST$L_BLI_BLKVEC_UNITS]);
    BLIUNIT(DSTPTR[DST$V_BLI_BLKVEC_UNIT_SIZE]);
    DBG$PRINT(UPLIT BYTE(%ASCIC '']), 0);
  END;

[INRANGE, OUTRANGE]:
  BEGIN
    DBG$PRINT(UPLIT BYTE(%ASCIC ' invalid structure'), 0);
  END;
```



```
: 2030      2152  2
: 2031      2153  2
: 2032      2154  2
: 2033      2155  2
: 2034      2156  2
: 2035      2157  2
: 2036      2158  2
: 2037      2159  2
: 2038      2160  2
: 2039      2161  2
: 2040      2162  2
: 2041      2163  2
: 2042      2164  2
: 2043      2165  1

      TES;
      IF .DSTPTR[DST$B_BLI_FORMAL] EQL 1
      THEN
        DBG$PRINT(UPLIT BYTE(%ASCIC', formal'), 0);
      IF .TRAIL2[DST$L_BLI_SIZE] NEQ 0
      THEN
        DBG$PRINT(UPLIT BYTE(%ASCIC', size: !UL bytes'), .TRAIL2[DST$L_BLI_SIZE]);
      RETURN 0;
      END;
```

```
.PSECT DBGSPLIT,NOWRT, SHR, PIC,0

      66 65 72 20 2C 05 0067F P.ADQ: .ASCII <5>\, ref\
      2C 01 00685 P.ADR: .ASCII <1>\,\
75 72 74 73 20 64 6C 65 69 66 20 6F 6E 20 13 00687 P.ADS: .ASCII <19>\ no field structure\
      65 72 75 74 63 00696
      2C 4C 55 21 5B 72 6F 74 63 65 76 20 0C 00698 P.ADT: .ASCII <12>\ vector[!UL,\
      65 74 79 62 04 006A8 P.ADU: .ASCII <4>\byte\
      64 72 6F 77 04 006AD P.ADV: .ASCII <4>\word\
      67 6E 6F 6C 04 006B2 P.ADW: .ASCII <4>\long\
      4C 53 21 03 006B7 P.ADX: .ASCII <3>\!SL\
      5D 64 65 6E 67 69 73 2C 08 006BB P.ADY: .ASCII <8>\,signed]\
      5D 01 006C4 P.ADZ: .ASCII <1>\]\
4C 55 21 5B 72 6F 74 63 65 76 74 69 62 20 0F 006C6 P.AEA: .ASCII <15>\ bitvector[!UL]\
      5D 006D5
      2C 4C 55 21 5B 6B 63 6F 6C 62 20 0B 006D6 P.AEB: .ASCII <11>\ block[!UL,\
      65 74 79 62 04 006E2 P.AEC: .ASCII <4>\byte\
      64 72 6F 77 04 006E7 P.AED: .ASCII <4>\word\
      67 6E 6F 6C 04 006EC P.AEE: .ASCII <4>\long\
      4C 53 21 03 006F1 P.AEF: .ASCII <3>\!SL\
      5D 01 006F5 P.AEG: .ASCII <1>\]\
21 5B 72 6F 74 63 65 76 6B 63 6F 6C 62 20 15 006F7 P.AEH: .ASCII <21>\ blockvector[!UL,!UL,\
      2C 4C 55 21 2C 4C 55 00706
      65 74 79 62 04 0070D P.AEI: .ASCII <4>\byte\
      64 72 6F 77 04 00712 P.AEJ: .ASCII <4>\word\
      67 6E 6F 6C 04 00717 P.AEK: .ASCII <4>\long\
      4C 53 21 03 0071C P.AEL: .ASCII <3>\!SL\
      5D 01 00720 P.AEM: .ASCII <1>\]\
63 75 72 74 73 20 64 69 6C 61 76 6E 69 20 12 00722 P.AEN: .ASCII <18>\ invalid structure\
      65 72 75 74 00731
      6C 61 6D 72 6F 66 20 2C 08 00735 P.AEO: .ASCII <8>\, formal\
79 62 20 4C 55 21 20 3A 65 7A 69 73 20 2C 11 0073E P.AEP: .ASCII <17>\, size: !UL bytes\
      73 65 74 0074D
```

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

00FC 00000 GET\_BLISS\_SPECIAL\_CASES:  
.WORD Save R2,R3,R4,R5,R6,R7

: 2038



|      |    |      |      |           |      |       |       |        |                       |      |
|------|----|------|------|-----------|------|-------|-------|--------|-----------------------|------|
|      |    |      | 57   | 00000000G | 00   | 9E    | 00002 | MOVAB  | DBG\$PRINT, R7        |      |
|      |    |      | 56   | 000000000 | EF   | 9E    | 00009 | MOVAB  | P.ADQ, R6             |      |
|      |    |      | 52   | 04        | AC   | D0    | 00010 | MOVL   | DSTPTR, R2            | 2080 |
|      |    |      | 50   | 02        | A2   | 9A    | 00014 | MOVZBL | 2(R2), R0             |      |
|      |    |      | 50   | 03        | A042 | 9E    | 00018 | MOVAB  | 3(R0)[R2], TRAIL1     |      |
|      |    |      | 51   | 04        | A0   | 9A    | 0001D | MOVZBL | 4(Trail1), R1         | 2081 |
|      |    |      | 54   | 05        | A140 | 9E    | 00021 | MOVAB  | 5(R1)[Trail1], TRAIL2 |      |
|      |    |      |      | 05        | A2   | 95    | 00026 | TSTB   | 5(R2)                 | 2082 |
|      |    |      |      |           | 06   | 18    | 00029 | BGEQ   | 1\$                   |      |
|      |    |      |      |           | 7E   | D4    | 0002B | CLRL   | -(SP)                 | 2084 |
|      |    |      |      |           | 56   | DD    | 0002D | PUSHL  | R6                    |      |
|      |    |      |      |           | 05   | 11    | 0002F | BRB    | 2\$                   |      |
|      |    |      |      |           | 7E   | D4    | 00031 | CLRL   | -(SP)                 | 2086 |
|      |    |      |      | 06        | A6   | 9F    | 00033 | PUSHAB | P.ADR                 |      |
|      |    |      | 67   | 02        | FB   | 00036 | 2\$:  | CALLS  | #2, DBG\$PRINT        |      |
| 55   | 05 | A2   | 03   | 00        | EF   | 00039 | 2\$:  | EXTZV  | #0, #3, 5(R2), R5     | 2091 |
| 006F |    | 0067 | 00   | 55        | CF   | 0003F |       | CASEL  | R5, #0, #4            |      |
|      |    |      | 0019 | 0012      |      | 00043 | 3\$:  | .WORD  | 4\$-3\$,-             |      |
|      |    |      |      | 00B1      |      | 0004B |       |        | 5\$-3\$,-             |      |
|      |    |      |      |           |      |       |       |        | 11\$-3\$,-            |      |
|      |    |      |      |           |      |       |       |        | 13\$-3\$,-            |      |
|      |    |      |      |           |      |       |       |        | 19\$-3\$              |      |
|      |    |      |      |           | 7E   | D4    | 0004D | CLRL   | -(SP)                 | 2150 |
|      |    |      | 00A3 | C6        | 9F   | 0004F |       | PUSHAB | P.AEN                 |      |
|      |    |      |      | 5B        | 11   | 00053 |       | BRB    | 12\$                  |      |
|      |    |      |      | 7E        | D4   | 00055 | 4\$:  | CLRL   | -(SP)                 | 2099 |
|      |    |      | 08   | A6        | 9F   | 00057 |       | PUSHAB | P.ADS                 |      |
|      |    |      |      | 54        | 11   | 0005A |       | BRB    | 12\$                  |      |
|      |    |      | 06   | A2        | DD   | 0005C | 5\$:  | PUSHL  | 6(R2)                 | 2107 |
|      |    |      | 1C   | A6        | 9F   | 0005F |       | PUSHAB | P.ADT                 |      |
|      |    |      | 67   | 02        | FB   | 00062 |       | CALLS  | #2, DBG\$PRINT        |      |
| 53   | 0A | A2   | 04   | 00        | EF   | 00065 |       | EXTZV  | #0, #4, 10(R2), R3    | 2108 |
|      |    |      | 01   | 53        | D1   | 0006B |       | CMPL   | R3, #1                |      |
|      |    |      |      | 07        | 12   | 0006E |       | BNEQ   | 6\$                   |      |
|      |    |      |      | 7E        | D4   | 00070 |       | CLRL   | -(SP)                 |      |
|      |    |      | 29   | A6        | 9F   | 00072 |       | PUSHAB | P.ADU                 |      |
|      |    |      |      | 1D        | 11   | 00075 |       | BRB    | 9\$                   |      |
|      |    |      | 02   | 53        | D1   | 00077 | 6\$:  | CMPL   | R3, #2                |      |
|      |    |      |      | 07        | 12   | 0007A |       | BNEQ   | 7\$                   |      |
|      |    |      |      | 7E        | D4   | 0007C |       | CLRL   | -(SP)                 |      |
|      |    |      | 2E   | A6        | 9F   | 0007E |       | PUSHAB | P.ADV                 |      |
|      |    |      |      | 11        | 11   | 00081 |       | BRB    | 9\$                   |      |
|      |    |      | 04   | 53        | D1   | 00083 | 7\$:  | CMPL   | R3, #4                |      |
|      |    |      |      | 07        | 12   | 00086 |       | BNEQ   | 8\$                   |      |
|      |    |      |      | 7E        | D4   | 00088 |       | CLRL   | -(SP)                 |      |
|      |    |      | 33   | A6        | 9F   | 0008A |       | PUSHAB | P.ADW                 |      |
|      |    |      |      | 05        | 11   | 0008D |       | BRB    | 9\$                   |      |
|      |    |      |      | 53        | DD   | 0008F | 8\$:  | PUSHL  | R3                    |      |
|      |    |      | 38   | A6        | 9F   | 00091 |       | PUSHAB | P.ADX                 |      |
|      |    |      |      | 02        | FB   | 00094 | 9\$:  | CALLS  | #2, DBG\$PRINT        |      |
|      |    |      | 67   | 04        | E1   | 00097 |       | BBC    | #4, 10(R2), 10\$      | 2109 |
|      | 0A | A2   |      | 7E        | D4   | 0009C |       | CLRL   | -(SP)                 | 2111 |
|      |    |      |      | A6        | 9F   | 0009E |       | PUSHAB | P.ADY                 |      |
|      |    |      |      | 4F        | 11   | 000A1 |       | BRB    | 18\$                  |      |
|      |    |      |      | 7E        | D4   | 000A3 | 10\$: | CLRL   | -(SP)                 | 2113 |
|      |    |      | 45   | A6        | 9F   | 000A5 |       | PUSHAB | P.ADZ                 |      |
|      |    |      |      | 48        | 11   | 000A8 |       | BRB    | 18\$                  |      |



|  |  |    |      |    |       |       |        |                    |        |
|--|--|----|------|----|-------|-------|--------|--------------------|--------|
|  |  | 06 | A2   | DD | 000AA | 11\$: | PUSHL  | 6(R2)              | : 2122 |
|  |  | 47 | A6   | 9F | 000AD |       | PUSHAB | P.AEA              | :      |
|  |  |    | 40   | 11 | 000B0 | 12\$: | BRB    | 18\$               | :      |
|  |  | 06 | A2   | DD | 000B2 | 13\$: | PUSHL  | 6(R2)              | : 2131 |
|  |  | 57 | A6   | 9F | 000B5 |       | PUSHAB | P.AEB              | : 2130 |
|  |  |    | 02   | FB | 000B8 |       | CALLS  | #2, DBG\$PRINT     | :      |
|  |  | 04 | 00   | EF | 000BB |       | EXTZV  | #0, #4, 10(R2), R3 | : 2132 |
|  |  | 01 | 53   | D1 | 000C1 |       | CMPL   | R3, #1             | :      |
|  |  |    | 07   | 12 | 000C4 |       | BNEQ   | 14\$               | :      |
|  |  |    | 7E   | D4 | 000C6 |       | CLRL   | -(SP)              | :      |
|  |  |    | 63   | A6 | 9F    | 000C8 | PUSHAB | P.AEC              | :      |
|  |  |    | 1D   | 11 | 000CB |       | BRB    | 17\$               | :      |
|  |  | 02 | 53   | D1 | 000CD | 14\$: | CMPL   | R3, #2             | :      |
|  |  |    | 07   | 12 | 000D0 |       | BNEQ   | 15\$               | :      |
|  |  |    | 7E   | D4 | 000D2 |       | CLRL   | -(SP)              | :      |
|  |  |    | 68   | A6 | 9F    | 000D4 | PUSHAB | P.AED              | :      |
|  |  |    | 11   | 11 | 000D7 |       | BRB    | 17\$               | :      |
|  |  | 04 | 53   | D1 | 000D9 | 15\$: | CMPL   | R3, #4             | :      |
|  |  |    | 07   | 12 | 000DC |       | BNEQ   | 16\$               | :      |
|  |  |    | 7E   | D4 | 000DE |       | CLRL   | -(SP)              | :      |
|  |  |    | 6D   | A6 | 9F    | 000E0 | PUSHAB | P.AEE              | :      |
|  |  |    | 05   | 11 | 000E3 |       | BRB    | 17\$               | :      |
|  |  |    | 53   | DD | 000E5 | 16\$: | PUSHL  | R3                 | :      |
|  |  |    | 72   | A6 | 9F    | 000E7 | PUSHAB | P.AEF              | :      |
|  |  | 67 | 02   | FB | 000EA | 17\$: | CALLS  | #2, DBG\$PRINT     | :      |
|  |  |    | 7E   | D4 | 000ED |       | CLRL   | -(SP)              | : 2133 |
|  |  |    | 76   | A6 | 9F    | 000EF | PUSHAB | P.AEG              | :      |
|  |  |    | 44   | 11 | 000F2 | 18\$: | BRB    | 24\$               | :      |
|  |  | 7E | 06   | A2 | 7D    | 000F4 | MOVQ   | 6(R2), -(SP)       | : 2142 |
|  |  |    | 78   | A6 | 9F    | 000F8 | PUSHAB | P.AEH              | : 2141 |
|  |  | 67 | 03   | FB | 000FB |       | CALLS  | #3, DBG\$PRINT     | :      |
|  |  | 53 | A2   | 9A | 000FE |       | MOVZBL | 14(R2), R3         | : 2144 |
|  |  | 01 | 53   | 91 | 00102 |       | CMPB   | R3, #1             | :      |
|  |  |    | 08   | 12 | 00105 |       | BNEQ   | 20\$               | :      |
|  |  |    | 7E   | D4 | 00107 |       | CLRL   | -(SP)              | :      |
|  |  |    | 008E | C6 | 9F    | 00109 | PUSHAB | P.AEI              | :      |
|  |  |    | 20   | 11 | 0010D |       | BRB    | 23\$               | :      |
|  |  | 02 | 53   | 91 | 0010F | 20\$: | CMPB   | R3, #2             | :      |
|  |  |    | 08   | 12 | 00112 |       | BNEQ   | 21\$               | :      |
|  |  |    | 7E   | D4 | 00114 |       | CLRL   | -(SP)              | :      |
|  |  |    | 0093 | C6 | 9F    | 00116 | PUSHAB | P.AEJ              | :      |
|  |  |    | 13   | 11 | 0011A |       | BRB    | 23\$               | :      |
|  |  | 04 | 53   | 91 | 0011C | 21\$: | CMPB   | R3, #4             | :      |
|  |  |    | 08   | 12 | 0011F |       | BNEQ   | 22\$               | :      |
|  |  |    | 7E   | D4 | 00121 |       | CLRL   | -(SP)              | :      |
|  |  |    | 0098 | C6 | 9F    | 00123 | PUSHAB | P.AEK              | :      |
|  |  |    | 06   | 11 | 00127 |       | BRB    | 23\$               | :      |
|  |  |    | 53   | DD | 00129 | 22\$: | PUSHL  | R3                 | :      |
|  |  |    | 009D | C6 | 9F    | 0012B | PUSHAB | P.AEL              | :      |
|  |  | 67 | 02   | FB | 0012F | 23\$: | CALLS  | #2, DBG\$PRINT     | :      |
|  |  |    | 7E   | D4 | 00132 |       | CLRL   | -(SP)              | : 2145 |
|  |  |    | 00A1 | C6 | 9F    | 00134 | PUSHAB | P.AEM              | :      |
|  |  | 67 | 02   | FB | 00138 | 24\$: | CALLS  | #2, DBG\$PRINT     | :      |
|  |  | 01 | A2   | 91 | 0013B |       | CMPB   | 3(R2), #1          | : 2155 |
|  |  |    | 09   | 12 | 0013F |       | BNEQ   | 25\$               | :      |
|  |  |    | 7E   | D4 | 00141 |       | CLRL   | -(SP)              | : 2157 |
|  |  |    | 00B6 | C6 | 9F    | 00143 | PUSHAB | P.AEO              | :      |



DBGSYMBOL  
V04-000

G 1  
16-Sep-1984 02:39:38 VAX-11 Bliss-32 V4.0-742  
14-Sep-1984 12:17:52 [DEBUG.SRC]DBGSYMBOL.B32;1

Page 69  
(12)

|    |      |    |       |       |        |                |   |      |
|----|------|----|-------|-------|--------|----------------|---|------|
| 67 |      | 02 | FB    | 00147 | CALLS  | #2, DBG\$PRINT | : |      |
|    |      | 64 | D5    | 0014A | TSTL   | (TRAIL2)       | : | 2159 |
|    |      | 09 | 13    | 0014C | BEQL   | 26\$           | : |      |
|    |      | 64 | DD    | 0014E | PUSHL  | (TRAIL2)       | : | 2161 |
|    | 00BF | C6 | 9F    | 00150 | PUSHAB | P.AEP          | : |      |
| 67 |      | 02 | FB    | 00154 | CALLS  | #2, DBG\$PRINT | : |      |
|    |      | 04 | 00157 | 26\$: | RET    |                | : | 2165 |

; Routine Size: 344 bytes, Routine Base: DBG\$CODE + 0C3A



```
2045 2166 1 ROUTINE GET_VARIANT(VARIANT_SYMID): NOVALUE =
2046 2167 1
2047 2168 1 FUNCTION
2048 2169 1     Get information from the RST for a variant.
2049 2170 1
2050 2171 1 INPUTS
2051 2172 1     VARIANT_SYMID - One of the Record Component RST entries in RST Entry
2052 2173 1                     for Data Type.
2053 2174 1
2054 2175 1 OUTPUTS
2055 2176 1     None.
2056 2177 1
2057 2178 1
2058 2179 2 BEGIN
2059 2180 2
2060 2181 2 MAP
2061 2182 2     VARIANT_SYMID: REF RST$ENTRY;    ! Pointer to RST entry for variant
2062 2183 2
2063 2184 2 LOCAL
2064 2185 2     BITSIZE,                          ! The size in bits of the given item
2065 2186 2     COMPVECPTR: REF VECTOR[.LONG],    ! Pointer to a vector of variant
2066 2187 2                                     ! component SYMIDs and tag values
2067 2188 2     DSTPTR: REF DST$RECORD,             ! Pointer to DST entry
2068 2189 2     ELTVECPTR,                        ! Pointer to a vector of field SYMIDs
2069 2190 2     LENGTH,                          ! Length of RST entry
2070 2191 2     LIST_ID: REF RST$VAR_ENTRY,        ! Pointer to variant component RST entry
2071 2192 2     NAME: REF VECTOR[.BYTE],          ! Pointer to counted ASCII string
2072 2193 2     NCOMPS,                          ! The number of variant components
2073 2194 2     NELTS,                          ! The number of fields
2074 2195 2     NTAGS,                          ! The number of tag blocks
2075 2196 2     SIZE,                          ! The size in bits of this variant
2076 2197 2     TAG: REF RST$ENTRY,                ! SYMID of the variant tag field
2077 2198 2     TAGVECPTR: REF VECTOR[.LONG];    ! Pointer to a vector of tag value ranges
2078 2199 2
2079 2200 2
2080 2201 2
2081 2202 2 DBG$NEWLINE();
2082 2203 2 DBG$STA_TYP VARIANT(.VARIANT_SYMID, NCOMPS, COMPVECPTR, TAG, BITSIZE);
2083 2204 2 DBG$PRINT CONTROL(DBG$K_PRTSET_RLMARGIN, +2);
2084 2205 2 DBG$PRINT(UPBIT BYTE(%ASCIC 'with !UL '), .NCOMPS);
2085 2206 2 LENGTH = DBG$TRANSLATE_KIND(.VARIANT_SYMID);
2086 2207 2 NAME = DBG$GET_DST_NAME(.TAG[RST$L_DSTPTR]);
2087 2208 2 IF .NAME[0] NEQ 0
2088 2209 2 THEN
2089 2210 2     DBG$PRINT(UPBIT BYTE(%ASCIC ', tag: !AC'), .NAME)
2090 2211 2 ELSE
2091 2212 2     DBG$PRINT(UPBIT BYTE(%ASCIC ', tag: anonymous'), 0);
2092 2213 2
2093 2214 2 ITEM SIZE(BITSIZE);
2094 2215 2 DBG$NEWLINE();
2095 2216 2 INCR I FROM 0 TO .NCOMPS - 1 DO
2096 2217 2     BEGIN
2097 2218 2     LIST_ID = .COMPVECPTR[I];
2098 2219 2     DBG$STA_TYP VARIANT COMP(.LIST_ID, NELTS, ELTVECPTR, NTAGS,
2099 2220 2     TAGVECPTR, SIZE);
2100 2221 2     DBG$PRINT(UPBIT BYTE(%ASCIC 'variant !UL:'), .I + 1);
2101 2222 2     DBG$PRINT(UPBIT BYTE(%ASCIC ' !UL fields'), .NELTS);
```



```
: 2102      2223      3      ITEM_SIZE(SIZE);
: 2103      2224      3      DBG$NEWLINE();
: 2104      2225      3      END;
: 2105      2226      3
: 2106      2227      3      DBG$ PRINT FLAG = FALSE;
: 2107      2228      3      IF .DBG$_QUALIFIER[SYMBOL_RST]
: 2108      2229      3      THEN
: 2109      2230      3          DBG$DUMP_HEX(.VARIANT SYMID, .LENGTH * 4, .VARIANT SYMID,
: 2110      2231      3              UPLIT BYTE(%ASCIC '          variant RST '));
: 2111      2232      3
: 2112      2233      3      DSTPTR = .VARIANT SYMID[RST$L DSTPTR];
: 2113      2234      3      IF .DBG$_QUALIFIER[SYMBOL_DST]
: 2114      2235      3      THEN
: 2115      2236      3          DBG$DUMP_HEX(.DSTPTR, .DSTPTR[DST$B_LENGTH] + 1, .DSTPTR,
: 2116      2237      3              UPLIT BYTE(%ASCIC '          variant DST '));
: 2117      2238      3
: 2118      2239      3      DBG$PRINT_CONTROL(DBG$_PRTSET_RLMARGIN, -2);
: 2119      2240      3      RETURN 0;
: 2120      2241      3
: 2121      2242      1      END;
```

```
.PSECT DBG$PLIT,NOWRT, SHR, PIC,0
6F 6D 79 6E 43 20 4C 55 21 20 68 74 69 77 09 00750 P.AEQ: .ASCII <9>\with !UL \
6F 6E 41 21 20 3A 67 61 74 20 2C 0A 0075A P.AER: .ASCII <10>\, tag: !AC\
6F 6E 61 20 3A 67 61 74 20 2C 10 00765 P.AES: .ASCII <16>\, tag: anonymous\
73 75 00774
20 3A 65 7A 69 73 20 2C 08 00776 P.AET: .ASCII <8>\, size: \
73 74 69 62 20 4C 55 21 08 0077F P.AEU: .ASCII <8>\!UL bits\
73 65 74 79 62 20 4C 55 21 09 00788 P.AEV: .ASCII <9>\!UL bytes\
73 74 69 62 20 4C 55 21 09 00792 P.AEW: .ASCII <9>\!UL bits\
3A 4C 55 21 20 74 6E 61 69 72 61 76 0C 0079C P.AEX: .ASCII <12>\variant !UL:\
73 64 6C 65 69 66 20 4C 55 21 20 0B 007A9 P.AEY: .ASCII <11>\!UL fields\
73 65 74 79 62 20 4C 55 21 08 007B5 P.AEZ: .ASCII <8>\, size: \
73 74 69 62 20 4C 55 21 09 007BE P.AFA: .ASCII <8>\!UL bits\
6E 61 69 72 61 76 20 20 20 20 20 20 20 14 007C7 P.AFB: .ASCII <9>\!UL bytes\
73 74 69 62 20 4C 55 21 09 007D1 P.AFC: .ASCII <9>\!UL bits\
6E 61 69 72 61 76 20 20 20 20 20 20 20 14 007DB P.AFD: .ASCII <20>\          variant RST \
20 54 53 52 20 74 007EA
6E 61 69 72 61 76 20 20 20 20 20 20 20 14 007F0 P.AFE: .ASCII <20>\          variant DST \
20 54 53 44 20 74 007FF
```

```
.PSECT DBG$CODE,NOWRT, SHR, PIC,0
OFFC 00000 GET_VARIANT:
5B 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11
5A 00000000' EF 9E 00009 MOVAB DBG$PRINT, R11
5E 24 C2 00010 MOVAB P.AEQ, R10
00000000G 00 00 FB 00013 SUBL2 #36, SP
5E DD 0001A CALLS #0, DBG$NEWLINE
08 AE 9F 0001C PUSHL SP
10 AE 9F 0001F PUSHAB TAG
PUSHAB COMVECPTR
```

2166

2202  
2203



|    |      |           |    |    |    |    |       |        |                          |                         |      |
|----|------|-----------|----|----|----|----|-------|--------|--------------------------|-------------------------|------|
|    |      | 57        | 04 | 18 | AE | 9F | 00022 | PUSHAB | NCOMPS                   |                         |      |
|    |      |           |    | 04 | AC | D0 | 00025 | MOVL   | VARIANT_SYMID, R7        |                         |      |
|    |      |           |    |    | 57 | DD | 00029 | PUSHL  | R7                       |                         |      |
|    |      | 00000000G | 00 |    | 05 | FB | 0002B | CALLS  | #5, DBG\$STA_TYP_VARIANT |                         |      |
|    |      |           |    |    | 02 | DD | 00032 | PUSHL  | #2                       |                         | 2204 |
|    |      |           |    |    | 02 | DD | 00034 | PUSHL  | #2                       |                         |      |
|    |      | 00000000G | 00 |    | 02 | FB | 00036 | CALLS  | #2, DBG\$PRINT_CONTROL   |                         |      |
|    |      |           |    | 0C | AE | DD | 0003D | PUSHL  | NCOMPS                   |                         | 2205 |
|    |      |           |    |    | 5A | DD | 00040 | PUSHL  | R10                      |                         |      |
|    |      | 68        |    |    | 02 | FB | 00042 | CALLS  | #2, DBG\$PRINT           |                         |      |
|    |      |           |    |    | 57 | DD | 00045 | PUSHL  | R7                       |                         | 2206 |
|    | FA9E | CF        |    |    | 01 | FB | 00047 | CALLS  | #1, DBG\$TRANSLATE_KIND  |                         |      |
|    |      | 59        |    |    | 50 | D0 | 0004C | MOVL   | R0, LENGTH               |                         |      |
|    |      | 50        |    | 04 | AE | D0 | 0004F | MOVL   | TAG, R0                  |                         | 2207 |
|    |      |           |    | 0C | A0 | DD | 00053 | PUSHL  | 12(R0)                   |                         |      |
|    |      | 00000000G | 00 |    | 01 | FB | 00056 | CALLS  | #1, DBG\$GET_DST_NAME    |                         |      |
|    |      |           |    |    | 60 | 95 | 0005D | TSTB   | (NAME)                   |                         | 2208 |
|    |      |           |    |    | 07 | 13 | 0005F | BEQL   | 1\$                      |                         |      |
|    |      |           |    |    | 50 | DD | 00061 | PUSHL  | NAME                     |                         | 2210 |
|    |      |           |    | 0A | AA | 9F | 00063 | PUSHAB | P.AER                    |                         |      |
|    |      |           |    |    | 05 | 11 | 00066 | BRB    | 2\$                      |                         |      |
|    |      |           |    |    | 7E | D4 | 00068 | CLRL   | -(SP)                    |                         | 2212 |
|    |      |           |    | 15 | AA | 9F | 0006A | PUSHAB | P.AES                    |                         |      |
|    |      | 68        |    |    | 02 | FB | 0006D | CALLS  | #2, DBG\$PRINT           |                         |      |
|    |      | 52        |    |    | 6E | D0 | 00070 | MOVL   | BITSIZE, R2              |                         | 2214 |
|    |      |           |    |    | 3C | 13 | 00073 | BEQL   | 6\$                      |                         |      |
|    | 53   | 52        |    |    | 08 | C7 | 00075 | DIVL3  | #8, R2, BYTES            |                         |      |
| 54 | 52   | 03        |    |    | 00 | EF | 00079 | EXTZV  | #0, #3, R2, BITS         |                         |      |
|    |      |           |    |    | 7E | D4 | 0007E | CLRL   | -(SP)                    |                         |      |
|    |      |           |    | 26 | AA | 9F | 00080 | PUSHAB | P.AET                    |                         |      |
|    |      | 68        |    |    | 02 | FB | 00083 | CALLS  | #2, DBG\$PRINT           |                         |      |
|    |      |           |    |    | 55 | D4 | 00086 | CLRL   | R5                       |                         |      |
|    |      |           |    |    | 54 | D5 | 00088 | TSTL   | BITS                     |                         |      |
|    |      |           |    |    | 0E | 13 | 0008A | BEQL   | 3\$                      |                         |      |
|    |      |           |    |    | 55 | D6 | 0008C | INCL   | R5                       |                         |      |
|    |      | 20        |    |    | 52 | D1 | 0008E | CMPL   | R2, #32                  |                         |      |
|    |      |           |    |    | 07 | 14 | 00091 | BGTR   | 3\$                      |                         |      |
|    |      |           |    |    | 52 | DD | 00093 | PUSHL  | R2                       |                         |      |
|    |      |           |    | 2F | AA | 9F | 00095 | PUSHAB | P.AEU                    |                         |      |
|    |      |           |    |    | 14 | 11 | 00098 | BRB    | 5\$                      |                         |      |
|    |      |           |    |    | 53 | D5 | 0009A | TSTL   | BYTES                    |                         |      |
|    |      |           |    |    | 08 | 13 | 0009C | BEQL   | 4\$                      |                         |      |
|    |      |           |    |    | 53 | DD | 0009E | PUSHL  | BYTES                    |                         |      |
|    |      |           |    | 38 | AA | 9F | 000A0 | PUSHAB | P.AEV                    |                         |      |
|    |      | 68        |    |    | 02 | FB | 000A3 | CALLS  | #2, DBG\$PRINT           |                         |      |
|    |      | 08        |    |    | 55 | E9 | 000A6 | BLBC   | R5, 6\$                  |                         |      |
|    |      |           |    |    | 54 | DD | 000A9 | PUSHL  | BITS                     |                         |      |
|    |      |           |    | 42 | AA | 9F | 000AB | PUSHAB | P.AEW                    |                         |      |
|    |      | 68        |    |    | 02 | FB | 000AE | CALLS  | #2, DBG\$PRINT           |                         |      |
|    |      | 00        |    |    | 00 | FB | 000B1 | CALLS  | #0, DBG\$NEWLINE         |                         | 2215 |
|    |      | 52        |    |    | 01 | CE | 000B8 | MNEGL  | #1, 1                    |                         | 2218 |
|    |      |           |    |    | 79 | 11 | 000BB | BRB    | 12\$                     |                         |      |
|    |      | 58        |    | 08 | BE | 42 | D0    | 000BD  | MOVL                     | @COMPVECPTR[1], LIST_ID |      |
|    |      |           |    | 10 | AE | 9F | 000C2 | PUSHAB | SIZE                     |                         | 2219 |
|    |      |           |    | 18 | AE | 9F | 000C5 | PUSHAB | TAGVECPTR                |                         |      |
|    |      |           |    | 20 | AE | 9F | 000C8 | PUSHAB | NTAGS                    |                         |      |
|    |      |           |    | 28 | AE | 9F | 000CB | PUSHAB | ELTVECPTR                |                         |      |



|    |           |    |      |             |        |                               |      |  |
|----|-----------|----|------|-------------|--------|-------------------------------|------|--|
|    |           |    | 30   | AE 9F 000CE | PUSHAB | NELTS                         |      |  |
|    |           |    | 58   | DD 000D1    | PUSHL  | LIST_ID                       |      |  |
|    | 00000000G | 00 | 06   | FB 000D3    | CALLS  | #6, DBG\$STA_TYP_VARIANT_COMP |      |  |
|    |           |    | 01   | A2 9F 000DA | PUSHAB | 1(1)                          | 2221 |  |
|    |           |    | 4C   | AA 9F 000DD | PUSHAB | P.AEX                         |      |  |
|    |           | 6B | 02   | FB 000E0    | CALLS  | #2, DBG\$PRINT                |      |  |
|    |           |    | 20   | AE DD 000E3 | PUSHL  | NELTS                         | 2222 |  |
|    |           |    | 59   | AA 9F 000E6 | PUSHAB | P.AEY                         |      |  |
|    |           | 6B | 02   | FB 000E9    | CALLS  | #2, DBG\$PRINT                |      |  |
|    |           | 53 | 10   | AE D0 000EC | MOVL   | SIZE, R3                      | 2223 |  |
|    |           |    | 3D   | 13 000F0    | BEQL   | 11\$                          |      |  |
|    | 54        | 53 | 08   | C7 000F2    | DIVL3  | #8, R3, BYTES                 |      |  |
| 55 | 53        | 03 | 00   | EF 000F6    | EXTZV  | #0, #3, R3, BITS              |      |  |
|    |           |    | 7E   | D4 000FB    | CLRL   | -(SP)                         |      |  |
|    |           |    | 65   | AA 9F 000FD | PUSHAB | P.AEZ                         |      |  |
|    |           | 6B | 02   | FB 00100    | CALLS  | #2, DBG\$PRINT                |      |  |
|    |           |    | 56   | D4 00103    | CLRL   | R6                            |      |  |
|    |           |    | 55   | D5 00105    | TSTL   | BITS                          |      |  |
|    |           |    | 0E   | 13 00107    | BEQL   | 8\$                           |      |  |
|    |           |    | 56   | D6 00109    | INCL   | R6                            |      |  |
|    |           | 20 | 53   | D1 0010B    | CMPL   | R3, #32                       |      |  |
|    |           |    | 07   | 14 0010E    | BGTR   | 8\$                           |      |  |
|    |           |    | 53   | DD 00110    | PUSHL  | R3                            |      |  |
|    |           |    | 6E   | AA 9F 00112 | PUSHAB | P.AFA                         |      |  |
|    |           |    | 15   | 11 00115    | BRB    | 10\$                          |      |  |
|    |           |    | 54   | D5 00117    | TSTL   | BYTES                         | 8\$: |  |
|    |           |    | 08   | 13 00119    | BEQL   | 9\$                           |      |  |
|    |           |    | 54   | DD 0011B    | PUSHL  | BYTES                         |      |  |
|    |           |    | 77   | AA 9F 0011D | PUSHAB | P.AFB                         |      |  |
|    |           | 6B | 02   | FB 00120    | CALLS  | #2, DBG\$PRINT                |      |  |
|    |           | 09 | 56   | E9 00123    | BLBC   | R6, 11\$                      | 9\$: |  |
|    |           |    | 55   | DD 00126    | PUSHL  | BITS                          |      |  |
|    |           |    | 0081 | CA 9F 00128 | PUSHAB | P.AFC                         |      |  |
|    |           | 6B | 02   | FB 0012C    | CALLS  | #2, DBG\$PRINT                |      |  |
|    | 00000000G | 00 | 00   | FB 0012F    | CALLS  | #0, DBG\$NEWLINE              | 2224 |  |
| 82 |           | 52 | 0C   | AE F2 00136 | AOBLSS | NCOMPS, 1, 7\$                | 2216 |  |
|    |           |    |      | EF D4 0013B | CLRL   | DBG\$ PRINT FLAG              | 2227 |  |
|    | 00000000' | EF | 04   | E1 00141    | BBC    | #4, DBG\$_QUALIFIER, 13\$     | 2228 |  |
| 11 |           |    | 008B | CA 9F 00149 | PUSHAB | P.AFD                         | 2231 |  |
|    |           |    | 57   | DD 0014D    | PUSHL  | R7                            | 2230 |  |
|    |           | 59 | 02   | 78 0014F    | ASHL   | #2, LENGTH, -(SP)             |      |  |
| 7E |           |    | 57   | DD 00153    | PUSHL  | R7                            |      |  |
|    | FC19      | CF | 04   | FB 00155    | CALLS  | #4, DBG\$DUMP_HEX             |      |  |
|    |           | 50 | A7   | D0 0015A    | MOVL   | 12(R7), DSTPTR                | 2233 |  |
| 12 | 00000000' | EF | 05   | E1 0015E    | BBC    | #5, DBG\$_QUALIFIER, 14\$     | 2234 |  |
|    |           |    | 00A0 | CA 9F 00166 | PUSHAB | P.AFE                         | 2237 |  |
|    |           |    | 50   | DD 0016A    | PUSHL  | DSTPTR                        | 2236 |  |
|    |           | 7E | 60   | 9A 0016C    | MOVZBL | (DSTPTR), -(SP)               |      |  |
|    |           |    | 6E   | D6 0016F    | INCL   | (SP)                          |      |  |
|    |           |    | 50   | DD 00171    | PUSHL  | DSTPTR                        |      |  |
|    | FBFB      | CF | 04   | FB 00173    | CALLS  | #4, DBG\$DUMP_HEX             |      |  |
|    |           | 7E | 02   | CE 00178    | MNEGL  | #2, -(SP)                     | 2239 |  |
|    |           |    | 02   | DD 0017B    | PUSHL  | #2                            |      |  |
|    | 00000000G | 00 | 02   | FB 0017D    | CALLS  | #2, DBG\$PRINT_CONTROL        |      |  |
|    |           |    | 04   | 00184       | RET    |                               | 2242 |  |

; Routine Size: 389 bytes, Routine Base: DBG\$CODE + 0D92



DBGSYMBOL  
V04-000

L 1  
16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

VAX-11 Bliss-32 V4.0-742  
[DEBUG.SRC]DBGSYMBOL.B32;1

Page 74  
(13)



```
2123 2243 1 ROUTINE SHOW_SYMBOL(RSTPTR, NCANDS): NOVALUE =
2124 2244 1
2125 2245 1 FUNCTION
2126 2246 1     This routine shows the kind of the symbol and the name of the symbol
2127 2247 1     in full pathname format. Additional information, such as data types
2128 2248 1     or address specifications, or RST, or DST is also shown if the
2129 2249 1     qualifiers are given.
2130 2250 1
2131 2251 1 INPUTS
2132 2252 1     RSTPTR - Symbol's RST entry
2133 2253 1     NCANDS - Keep track of the number of symbols is shown
2134 2254 1
2135 2255 1 OUTPUTS
2136 2256 1     None.
2137 2257 1
2138 2258 1 BEGIN
2139 2259 2
2140 2260 2 MAP
2141 2261 2     NCANDS: REF VECTOR[1],           ! Keep track of outputted symbols
2142 2262 2     RSTPTR: REF RST$ENTRY;         ! Selected candidate symbol RST entry
2143 2263 2
2144 2264 2 LOCAL
2145 2265 2     DSTPTR: REF DST$RECORD,         ! Pointer to DST entry
2146 2266 2     LANGUAGE: REF VECTOR[.BYTE],    ! Pointer to counted ASCII string
2147 2267 2     LENGTH,                          ! Length for RST entry
2148 2268 2     KIND,                            ! Symbol's kind field
2149 2269 2     OL_TRAILER:                      ! Overloaded symbol's DST trailer
2150 2270 2     REF DST$OVERLOAD TRLR,
2151 2271 2     OL_VECTOR: REF VECTOR[.LONG],    ! Vector of DST pointers for overloaded
2152 2272 2                                     symbols
2153 2273 2     PATHNAME,                        ! Pointer to returned pathname vector
2154 2274 2     POOLID,                         ! Temporary Memory Pool Stack ID
2155 2275 2     SIZE,                           ! Length for TYPE RST entry
2156 2276 2     SYMBOL_CNT,                     ! Symbol outputted count
2157 2277 2     SYMNAME,                        ! Pointer to a counted string
2158 2278 2     TYPEID: REF RST$ENTRY;          ! Pointer to TYPE RST entry
2159 2279 2
2160 2280 2
2161 2281 2
2162 2282 2
2163 2283 2 ! Abort this operation now if the Control-Y DEBUG flag is set.
2164 2284 2
2165 2285 2 $ABORT_ON_CONTROL_Y;
2166 2286 2
2167 2287 2
2168 2288 2 ! Everytime this routine is called, increment the count up by 1.
2169 2289 2
2170 2290 2 SYMBOL_CNT = .NCANDS[0];
2171 2291 2 SYMBOL_CNT = .SYMBOL_CNT + 1;
2172 2292 2
2173 2293 2
2174 2294 2 ! Initialization. Assume symbol does not have any type and the length
2175 2295 2 ! of the TYPE RST entry is zero.
2176 2296 2
2177 2297 2 TYPEID = 0;
2178 2298 2 SIZE = 0;
2179 2299 2
```



```
2180 2300 2
2181 2301
2182 2302
2183 2303
2184 2304
2185 2305
2186 2306
2187 2307
2188 2308
2189 2309
2190 2310
2191 2311
2192 2312
2193 2313
2194 2314
2195 2315
2196 2316
2197 2317
2198 2318
2199 2319
2200 2320
2201 2321
2202 2322
2203 2323
2204 2324
2205 2325
2206 2326
2207 2327
2208 2328
2209 2329
2210 2330
2211 2331
2212 2332
2213 2333
2214 2334
2215 2335
2216 2336
2217 2337
2218 2338
2219 2339
2220 2340
2221 2341
2222 2342
2223 2343
2224 2344
2225 2345
2226 2346
2227 2347
2228 2348
2229 2349
2230 2350
2231 2351
2232 2352
2233 2353
2234 2354
2235 2355
2236 2356 4

! Get symbol's kind field, and translate it into user readable form.
DBG$STA SYMKIND(.RSTPTR, KIND);
LENGTH = DBG$TRANSLATE_KIND(.RSTPTR);

! Push the tempory memory pointer.
POOLID = DBG$PUSH_TEMPMEM();

! Get the symbol's fully qualified pathname descriptor, and then call
! DBG$NPATHDESC TO_CS to translate the contents of the pathname descriptor
! into a printable form.
DBG$STA SYMPATHNAME(.RSTPTR, PATHNAME);
DBG$NPATHDESC TO_CS(.PATHNAME, SYMNAME);
DBG$PRINT(UPLIT BYTE(%ASCIC ' !AC'), .SYMNAME);

! Print out the Language used for module.
IF .KIND EQL RST$K_MODULE
THEN
  BEGIN
    DSTPTR = .RSTPTR[RST$L_DSTPTR];
    LANGUAGE = DBG$LANGUAGE(.DSTPTR[DST$L_MODBEG LANGUAGE]);
    DBG$PRINT(UPLIT BYTE(%ASCIC ' language !AC'), .LANGUAGE);
  END;

! Check to see if this symbol is global symbol. Mark it if so.
IF .RSTPTR[RST$V_GLOBAL]
THEN
  DBG$PRINT(UPLIT BYTE(%ASCIC ' (global)'), 0);

! Output symbol kind and name, or (global).
DBG$NEWLINE();

! Print the instances for an Overloaded symbol.
IF .KIND EQL RST$K_OVERLOAD
THEN
  BEGIN
    DBG$STA SYMPATHNAME(.RSTPTR[RST$L_UPSCOPEPTR], PATHNAME);
    DBG$NPATHDESC TO_CS(.PATHNAME, SYMNAME);
    DSTPTR = .RSTPTR[RST$L_DSTPTR];
    OL_TRAILER = DSTPTR[DST$L_OL_TRAILER] + .DSTPTR[DST$L_OL_NAME];
    OL_VECTOR = OL_TRAILER[DST$L_OL_VECTOR];
    INCR I FROM 0 TO .OL_TRAILER[DST$L_OL_COUNT] -1 DO
      BEGIN
        DBG$PRINT(UPLIT BYTE(%ASCIC ' overloaded instance !AC'), .SYMNAME);
```



```
2237 2357 4      DBG$PRINT(UPRIT BYTE ( %ASCIC'\\!AC' ),
2238 2358 4          DBG$GET_DST_NAME(DBG$RST_DST_PTR(.OL_VECTOR[.I] +
2239 2359 4          .DST$BEGIN_ADDR)));
2240 2360 4      DBG$NEWLINE();
2241 2361 4      END;
2242 2362 4
2243 2363 4      END;
2244 2364 4
2245 2365 4
2246 2366 4      ! Pop the temporary memory pointer.
2247 2367 4      !
2248 2368 4      DBG$POP_TEMP_MEM(.POOLID);
2249 2369 4
2250 2370 4
2251 2371 4      ! Output additional information according to the given command qualifier.
2252 2372 4      ! /ADDRESS -- Print the address specification for each selected symbol.
2253 2373 4
2254 2374 4      IF .DBG$_QUALIFIER[SYMBOL_ADDRESS] AND .KIND NEQ RST$K_MODULE
2255 2375 4      THEN
2256 2376 4          BEGIN
2257 2377 4              DBG$PRINT_CONTROL(DBG$K_PRTSET_LMARGIN, 4);
2258 2378 4              SHOW_SYMBOL_ADDRESS(.RSTPTR);
2259 2379 4              DBG$PRINT_CONTROL(DBG$K_PRTSET_LMARGIN, 0);
2260 2380 4          END;
2261 2381 4
2262 2382 4
2263 2383 4      ! /TYPE -- Print data or record component type information for each
2264 2384 4      ! selected symbol. Set the context needed for subsequent DST value
2265 2385 4      ! spec evaluation.
2266 2386 4
2267 2387 4      IF .DBG$ QUALIFIER[SYMBOL TYPE] AND
2268 2388 4      NOT .RSTPTR[RST$V GLOBAL] AND
2269 2389 4      (.KIND EQL RST$K_DATA OR .KIND EQL RST$K_TYPCOMP)
2270 2390 4      THEN
2271 2391 4          BEGIN
2272 2392 4              DBG$STA_SETCONTEXT(.RSTPTR);
2273 2393 4              DBG$ PRINT FLAG = TRUE;
2274 2394 4              DBG$PRINT_CONTROL(DBG$K_PRTSET_RLMARGIN, +4);
2275 2395 4              SHOW_SYMBOL_TYPE(.RSTPTR);
2276 2396 4              DBG$PRINT_CONTROL(DBG$K_PRTSET_RLMARGIN, -4);
2277 2397 4          END;
2278 2398 4
2279 2399 4
2280 2400 4      ! If data or record component symbol has TYPE field, get a bit more
2281 2401 4      ! TYPE RST information about this symbol for /RST and /DST.
2282 2402 4
2283 2403 4      IF .KIND EQL RST$K_DATA OR
2284 2404 4      .KIND EQL RST$K_TYPCOMP
2285 2405 4      THEN
2286 2406 4          IF .RSTPTR[RST$L_TYPEPTR] NEQ 0
2287 2407 4          THEN
2288 2408 4              BEGIN
2289 2409 4                  TYPEID = .RSTPTR[RST$L_TYPEPTR];
2290 2410 4                  SIZE = RST$K_TYPCOMPENT + .TYPEID[RST$L_TYPCOMPENT];
2291 2411 4              END;
2292 2412 4
2293 2413 4
```



```
! /RST -- Print each selected symbol's RST entry in hexadecimal.
! (for developers only.)
IF .DBG$_QUALIFIER[SYMBOL_RST]
THEN
  BEGIN
    DBGSDUMP_HEX(.RSTPTR, .LENGTH * 4, .RSTPTR,
      UPLIT BYTE(%ASCIC ' RST entry dump '));

    ! If the symbol has TYPE RST entry, dump it as well.
    IF .TYPEID NEQ 0 THEN DBGSDUMP_HEX(.TYPEID, .SIZE * 4, .TYPEID,
      UPLIT BYTE(%ASCIC ' TYPE RST entry '));
  END;

! /DST -- Print each selected symbol's DST entry in hexadecimal.
! (for developers only.)
IF .DBG$_QUALIFIER[SYMBOL_DST]
THEN
  BEGIN
    DSTPTR = .RSTPTR[RST$L_DSTPTR];
    DBGSDUMP_HEX(.DSTPTR, .DSTPTR[DST$B_LENGTH] + 1, .DSTPTR,
      UPLIT BYTE(%ASCIC ' DST entry dump '));

    ! Save the value of DSTPTR as '\'. This is useful for poking
    ! the DST in tests. It doesn't affect behavior that the user
    ! sees since /DST is under the control of developer 0.
    DBGSSAVE_VAL(DBG$MAKE_INTEGER_DESC(.DSTPTR));

    ! If the symbol has TYPE RST DST entry, dump it as well.
    IF .TYPEID NEQ 0
    THEN
      BEGIN
        DSTPTR = .TYPEID[RST$L_DSTPTR];
        DBGSDUMP_HEX(.DSTPTR, .DSTPTR[DST$B_LENGTH] + 1, .DSTPTR,
          UPLIT BYTE(%ASCIC ' TYPE DST entry '));
      END;
    END;

    ! Update the count.
    NCANDS[0] = .SYMBOL_CNT;
    RETURN 0;
  END;
```

.PSECT DBG\$PLIT,NOWRT, SHR, PIC,0



|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |        |        |                                |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|--------|--------|--------------------------------|
| 43 | 41 | 21 | 20 | 65 | 67 | 61 | 75 | 67 | 6E | 43 | 41 | 21 | 20 | 04 | 00805 | P.AFF: | .ASCII | <4>\ !AC\                      |
|    |    |    |    |    | 29 | 6C | 61 | 62 | 6F | 61 | 6C | 20 | 2C | 0E | 0080A | P.AFG: | .ASCII | <14>\, language !AC\           |
| 69 | 20 | 64 | 65 | 64 | 61 | 6F | 6C | 72 | 65 | 76 | 6F | 20 | 20 | 09 | 00819 | P.AFH: | .ASCII | <9>\ (global)\                 |
|    |    |    |    | 43 | 41 | 21 | 20 | 65 | 63 | 6E | 61 | 74 | 73 | 6E | 00823 | P.AFI: | .ASCII | <25>\ overloaded instance !AC\ |
|    |    |    |    |    |    |    |    |    |    | 43 | 41 | 21 | 5C | 04 | 00832 |        |        |                                |
| 20 | 79 | 72 | 74 | 6E | 65 | 20 | 54 | 53 | 52 | 20 | 20 | 20 | 20 | 14 | 0083D | P.AFJ: | .ASCII | <4><92>\ !AC\                  |
|    |    |    |    |    |    |    |    |    | 20 | 20 | 70 | 6D | 75 | 64 | 00842 | P.AFK: | .ASCII | <20>\ RST entry dump \         |
| 65 | 20 | 54 | 53 | 52 | 20 | 45 | 50 | 59 | 54 | 20 | 20 | 20 | 20 | 14 | 00851 |        |        |                                |
|    |    |    |    |    |    |    |    |    | 20 | 20 | 79 | 72 | 74 | 6E | 00857 | P.AFL: | .ASCII | <20>\ TYPE RST entry \         |
| 20 | 79 | 72 | 74 | 6E | 65 | 20 | 54 | 53 | 44 | 20 | 20 | 20 | 20 | 14 | 00866 | P.AFM: | .ASCII | <20>\ DST entry dump \         |
|    |    |    |    |    |    |    |    |    | 20 | 20 | 70 | 6D | 75 | 64 | 0087B |        |        |                                |
| 65 | 20 | 54 | 53 | 44 | 20 | 45 | 50 | 59 | 54 | 20 | 20 | 20 | 20 | 14 | 00881 | P.AFN: | .ASCII | <20>\ TYPE DST entry \         |
|    |    |    |    |    |    |    |    |    | 20 | 20 | 79 | 72 | 74 | 6E | 00890 |        |        |                                |

```
.EXTRN  DBG$GV_CONTROL
```

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

| Offset   | Symbol | Disassembly | Comment                              | Address |
|----------|--------|-------------|--------------------------------------|---------|
| 00000000 |        | WORD        | Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 | 2243    |
| 00000000 |        | SUBL2       | #16, SP                              | 2279    |
| 00000000 |        | BBC         | #1, DBG\$GV_CONTROL+1, 1\$           | 2290    |
| 00000000 |        | PUSHL       | #164072                              | 2291    |
| 00000000 |        | CALLS       | #1, LIB\$SIGNAL                      | 2297    |
| 00000000 |        | MOVL        | @NCANDS, SYMBOL_CNT                  | 2298    |
| 00000000 |        | INCL        | SYMBOL_CNT                           | 2303    |
| 00000000 |        | CLRL        | TYPEID                               | 2304    |
| 00000000 |        | CLRL        | SIZE                                 | 2309    |
| 00000000 |        | PUSHAB      | KIND                                 | 2316    |
| 00000000 |        | MOVL        | RSTPTR, R3                           | 2317    |
| 00000000 |        | PUSHL       | R3                                   | 2318    |
| 00000000 |        | CALLS       | #2, DBG\$STA_SYMKIND                 | 2323    |
| 00000000 |        | PUSHL       | R3                                   | 2326    |
| 00000000 |        | CALLS       | #1, DBG\$TRANSLATE_KIND              | 2327    |
| 00000000 |        | MOVL        | R0, LENGTH                           | 2328    |
| 00000000 |        | CALLS       | #0, DBG\$PUSH_TEMPMEM                | 2334    |
| 00000000 |        | MOVL        | R0, POOLID                           |         |
| 00000000 |        | PUSHAB      | PATHNAME                             |         |
| 00000000 |        | PUSHL       | R3                                   |         |
| 00000000 |        | CALLS       | #2, DBG\$STA_SYMPATHNAME             |         |
| 00000000 |        | PUSHAB      | SYMNAME                              |         |
| 00000000 |        | PUSHL       | PATHNAME                             |         |
| 00000000 |        | CALLS       | #2, DBG\$NPATHTDESC_TO_CS            |         |
| 00000000 |        | PUSHL       | SYMNAME                              |         |
| 00000000 |        | PUSHAB      | P.AFF                                |         |
| 00000000 |        | CALLS       | #2, DBG\$PRINT                       |         |
| 00000000 |        | MOVL        | KIND, R6                             |         |
| 00000000 |        | CMPL        | R6, #1                               |         |
| 00000000 |        | BNEQ        | 2\$                                  |         |
| 00000000 |        | MOVL        | 12(R3), DSTPTR                       |         |
| 00000000 |        | PUSHL       | 3(DSTPTR)                            |         |
| 00000000 |        | CALLS       | #1, DBG\$LANGUAGE                    |         |
| 00000000 |        | PUSHL       | LANGUAGE                             |         |
| 00000000 |        | PUSHAB      | P.AFG                                |         |
| 00000000 |        | CALLS       | #2, DBG\$PRINT                       |         |
| 00000000 |        | BLBC        | 21(R3), 3\$                          |         |



|              |      |           |    |    |       |        |                                        |                           |      |
|--------------|------|-----------|----|----|-------|--------|----------------------------------------|---------------------------|------|
|              |      |           | 7E | D4 | 0009B | CLRL   | -(SP)                                  | 2336                      |      |
|              |      | 00000000' | EF | 9F | 0009D | PUSHAB | P.AFH                                  |                           |      |
| 00000000G    | 00   |           | 02 | FB | 000A3 | CALLS  | #2, DBG\$PRINT                         | 2341                      |      |
| 00000000G    | 00   |           | 00 | FB | 000AA | CALLS  | #0, DBG\$NEWLINE                       | 2346                      |      |
|              | 0D   |           | 56 | D1 | 000B1 | CMPL   | R6, #13                                |                           |      |
|              |      |           | 76 | 12 | 000B4 | BNEQ   | 6\$                                    | 2349                      |      |
|              |      | 08        | AE | 9F | 000B6 | PUSHAB | PATHNAME                               |                           |      |
|              |      | 10        | A3 | DD | 000B9 | PUSHL  | 16(R3)                                 | 2350                      |      |
| 00000000G    | 00   |           | 02 | FB | 000BC | CALLS  | #2, DBG\$STA_SYMPATHNAME               |                           |      |
|              |      | 0C        | AE | 9F | 000C3 | PUSHAB | SYMPATHNAME                            | 2351                      |      |
|              |      | 0C        | AE | DD | 000C6 | PUSHL  | PATHNAME                               | 2352                      |      |
| 00000000G    | 00   |           | 02 | FB | 000C9 | CALLS  | #2, DBG\$NPATHDESC_TO_CS               |                           |      |
|              | 55   |           | 0C | A3 | DD    | 000D0  | MOVL                                   | 12(R3), DSTPTR            | 2353 |
|              | 50   |           | 02 | A5 | 9A    | 000D4  | MOVZBL                                 | 2(DSTPTR), R0             | 2354 |
|              | 50   |           | 03 | A0 | 9E    | 000D8  | MOVAB                                  | 3(R0)[DSTPTR], OL_TRAILER |      |
|              | 52   |           | 02 | A0 | 9E    | 000DD  | MOVAB                                  | 2(R0), OL_VECTOR          | 2355 |
|              | 58   |           |    | 60 | 3C    | 000E1  | MOVZWL                                 | (OL_TRAILER), R8          | 2356 |
|              | 54   |           |    | 01 | CE    | 000E4  | MNEGL                                  | #1, I                     |      |
|              |      |           | 3F | 11 | 000E7 | BRB    | 5\$                                    | 2357                      |      |
|              |      | 0C        | AE | DD | 000E9 | PUSHL  | SYMPATHNAME                            | 2358                      |      |
|              |      | 00000000' | EF | 9F | 000EC | PUSHAB | P.AFI                                  |                           |      |
| 00000000G    | 00   |           | 02 | FB | 000F2 | CALLS  | #2, DBG\$PRINT                         | 2359                      |      |
| 7E 00000000G | 6244 | 00000000G | 00 | C1 | 000F9 | ADDL3  | DST\$BEGIN_ADDR, (OL_VECTOR)[1], -(SP) | 2358                      |      |
| 00000000G    | 00   |           | 01 | FB | 00102 | CALLS  | #1, DBG\$RST_DST_PTR                   |                           |      |
|              |      |           | 50 | DD | 00109 | PUSHL  | R0                                     |                           |      |
| 00000000G    | 00   |           | 01 | FB | 0010B | CALLS  | #1, DBG\$GET_DST_NAME                  |                           |      |
|              |      |           | 50 | DD | 00112 | PUSHL  | R0                                     | 2357                      |      |
|              |      | 00000000' | EF | 9F | 00114 | PUSHAB | P.AFJ                                  |                           |      |
| 00000000G    | 00   |           | 02 | FB | 0011A | CALLS  | #2, DBG\$PRINT                         | 2360                      |      |
| 00000000G    | 00   |           | 00 | FB | 00121 | CALLS  | #0, DBG\$NEWLINE                       | 2354                      |      |
| BD 00000000G | 54   |           | 58 | F2 | 0012B | AOBLSS | R8, I, 4\$                             | 2368                      |      |
|              |      |           | 5B | DD | 0012C | PUSHL  | POOLID                                 |                           |      |
| 00000000G    | 00   |           | 01 | FB | 0012E | CALLS  | #1, DBG\$POP_TEMP MEM                  | 2374                      |      |
| 21 00000000' | EF   |           | 02 | E1 | 00135 | BBC    | #2, DBG\$_QUALIFIER, 7\$               |                           |      |
|              | 01   |           | 56 | D1 | 0013D | CMPL   | R6, #1                                 | 2377                      |      |
|              |      |           | 1C | 13 | 00140 | BEQL   | 7\$                                    |                           |      |
|              |      |           | 04 | DD | 00142 | PUSHL  | #4                                     | 2378                      |      |
| 00000000G    | 00   |           | 01 | DD | 00144 | PUSHL  | #1                                     |                           |      |
|              |      |           | 02 | FB | 00146 | CALLS  | #2, DBG\$PRINT_CONTROL                 | 2379                      |      |
|              |      |           | 53 | DD | 0014D | PUSHL  | R3                                     |                           |      |
| 0000V        | CF   |           | 01 | FB | 0014F | CALLS  | #1, SHOW_SYMBOL_ADDRESS                | 2387                      |      |
|              | 7E   |           | 01 | 7D | 00154 | MOVQ   | #1, -(SP)                              | 2388                      |      |
| 00000000G    | 00   |           | 02 | FB | 00157 | CALLS  | #2, DBG\$PRINT_CONTROL                 | 2389                      |      |
| 3C 00000000' | EF   |           | 01 | E1 | 0015E | BBC    | #1, DBG\$_QUALIFIER, 9\$               |                           |      |
|              | 38   |           | A3 | E8 | 00166 | BLBS   | 21(R3), 9\$                            |                           |      |
|              | 06   |           | 56 | D1 | 0016A | CMPL   | R6, #6                                 |                           |      |
|              |      |           | 05 | 13 | 0016D | BEQL   | 8\$                                    | 2392                      |      |
|              | 0A   |           | 56 | D1 | 0016F | CMPL   | R6, #10                                |                           |      |
|              |      |           | 2E | 12 | 00172 | BNEQ   | 9\$                                    |                           |      |
|              |      |           | 53 | DD | 00174 | PUSHL  | R3                                     | 2393                      |      |
| 00000000G    | 00   |           | 01 | FB | 00176 | CALLS  | #1, DBG\$STA_SETCONTEXT                |                           |      |
| 00000000'    | EF   |           | 01 | DD | 0017D | MOVL   | #1, DBG\$_PRINT_FLAG                   | 2394                      |      |
|              |      |           | 04 | DD | 00184 | PUSHL  | #4                                     |                           |      |
|              |      |           | 02 | DD | 00186 | PUSHL  | #2                                     |                           |      |
| 00000000G    | 00   |           | 02 | FB | 00188 | CALLS  | #2, DBG\$PRINT_CONTROL                 | 2395                      |      |
|              |      |           | 53 | DD | 0018F | PUSHL  | R3                                     |                           |      |
| 0000V        | CF   |           | 01 | FB | 00191 | CALLS  | #1, SHOW_SYMBOL_TYPE                   |                           |      |



|              |    |    |             |            |                            |                   |  |
|--------------|----|----|-------------|------------|----------------------------|-------------------|--|
| 7E           | 04 | CE | 00196       | MNEGL      | #4, -(SP)                  | 2396              |  |
|              | 02 | DD | 00199       | PUSHL      | #2                         |                   |  |
| 00000000G    | 00 | 02 | FB 0019B    | CALLS      | #2, DBG\$PRINT_CONTROL     |                   |  |
|              | 06 | 56 | D1 001A2    | 9\$: CMPL  | R6, #6                     | 2403              |  |
|              |    | 05 | 13 001A5    | BEQL       | 10\$                       |                   |  |
| 0A           |    | 56 | D1 001A7    | CMPL       | R6, #10                    | 2404              |  |
|              |    | 0E | 12 001AA    | BNEQ       | 11\$                       |                   |  |
|              | 18 | A3 | D5 001AC    | 10\$: TSTL | 24(R3)                     | 2406              |  |
|              |    | 09 | 13 001AF    | BEQL       | 11\$                       |                   |  |
|              | 18 | A3 | D0 001B1    | MOVL       | 24(R3), TYPEID             | 2409              |  |
| 5A 28        | A7 | 0B | C1 001B5    | ADDL3      | #11, 40(TYPEID), SIZE      | 2410              |  |
| 2B 00000000' | EF | 04 | E1 001BA    | 11\$: BBC  | #4, DBG\$QUALIFIER, 12\$   | 2417              |  |
|              |    | EF | 9F 001C2    | PUSHAB     | P.AFK                      | 2421              |  |
|              |    | 53 | DD 001C8    | PUSHL      | R3                         | 2420              |  |
| 7E           | 08 | AE | 02          | 78 001CA   | ASHL                       | #2, LENGTH, -(SP) |  |
|              |    | 53 | DD 001CF    | PUSHL      | R3                         |                   |  |
| FA18         | CF | 04 | FB 001D1    | CALLS      | #4, DBG\$DUMP_HEX          |                   |  |
|              |    | 57 | D5 001D6    | TSTL       | TYPEID                     | 2426              |  |
|              |    | 13 | 13 001D8    | BEQL       | 12\$                       |                   |  |
|              |    | EF | 9F 001DA    | PUSHAB     | P.AFL                      | 2427              |  |
|              |    | 57 | DD 001E0    | PUSHL      | TYPEID                     | 2426              |  |
| 7E           | 5A | 02 | 78 001E2    | ASHL       | #2, SIZE, -(SP)            |                   |  |
|              |    | 57 | DD 001E6    | PUSHL      | TYPEID                     |                   |  |
| FA01         | CF | 04 | FB 001E8    | CALLS      | #4, DBG\$DUMP_HEX          |                   |  |
| 46 00000000' | EF | 05 | E1 001ED    | 12\$: BBC  | #5, DBG\$QUALIFIER, 13\$   | 2434              |  |
|              |    | 55 | A3 D0 001F5 | MOVL       | 12(R3), DSTPTR             | 2437              |  |
|              |    | EF | 9F 001F9    | PUSHAB     | P.AFM                      | 2439              |  |
|              |    | 55 | DD 001FF    | PUSHL      | DSTPTR                     | 2438              |  |
| 7E           |    | 65 | 9A 00201    | MOVZBL     | (DSTPTR), -(SP)            |                   |  |
|              |    | 6E | D6 00204    | INCL       | (SP)                       |                   |  |
|              |    | 55 | DD 00206    | PUSHL      | DSTPTR                     |                   |  |
| F9E1         | CF | 04 | FB 00208    | CALLS      | #4, DBG\$DUMP_HEX          |                   |  |
|              |    | 55 | DD 0020D    | PUSHL      | DSTPTR                     | 2445              |  |
| 00000000G    | 00 | 01 | FB 0020F    | CALLS      | #1, DBG\$MAKE_INTEGER_DESC |                   |  |
|              |    | 50 | DD 00216    | PUSHL      | R0                         |                   |  |
| 00000000G    | 00 | 01 | FB 00218    | CALLS      | #1, DBG\$SAVE_VAL          |                   |  |
|              |    | 57 | D5 0021F    | TSTL       | TYPEID                     | 2449              |  |
|              |    | 18 | 13 00221    | BEQL       | 13\$                       |                   |  |
|              |    | A7 | D0 00223    | MOVL       | 12(TYPEID), DSTPTR         | 2452              |  |
|              |    | EF | 9F 00227    | PUSHAB     | P.AFN                      | 2454              |  |
|              |    | 55 | DD 0022D    | PUSHL      | DSTPTR                     | 2453              |  |
| 7E           |    | 65 | 9A 0022F    | MOVZBL     | (DSTPTR), -(SP)            |                   |  |
|              |    | 6E | D6 00232    | INCL       | (SP)                       |                   |  |
|              |    | 55 | DD 00234    | PUSHL      | DSTPTR                     |                   |  |
| F9B3         | CF | 04 | FB 00236    | CALLS      | #4, DBG\$DUMP_HEX          |                   |  |
| 08           | BC | 59 | D0 0023B    | 13\$: MOVL | SYMBOL_CNT, INCANDS        | 2462              |  |
|              |    | 04 | 0023F       | RET        |                            | 2465              |  |

; Routine Size: 576 bytes, Routine Base: DBG\$CODE + 0F17



```
2347 2466 1 ROUTINE SHOW_SYMBOL_ADDRESS(SYMID): NOVALUE =
2348 2467 1
2349 2468 1 FUNCTION
2350 2469 1     This routine takes a SYMID and shows the address information associated
2351 2470 1     with the input SYMID.
2352 2471 1
2353 2472 1 INPUTS
2354 2473 1     SYMID - The SYMID for the symbol.
2355 2474 1
2356 2475 1 OUTPUTS
2357 2476 1     None.
2358 2477 1
2359 2478 1
2360 2479 2 BEGIN
2361 2480 2
2362 2481 2 MAP
2363 2482 2     SYMID: REF RST$ENTRY;           ! Pointer to input symbol's RST entry
2364 2483 2
2365 2484 2 LOCAL
2366 2485 2     ADDRKIND,                     ! Symbol's address kind
2367 2486 2     ADDRPTR: VECTOR[2],           ! Symbol's address vector
2368 2487 2     BLITRLR: REF DST$BLI TRAILER1, ! Pointer to Bliss DST record trailer
2369 2488 2     BLIVALSPEC: BLOCK[8, BYTE]     ! Value Spec buffer for Bliss Special
2370 2489 2     FIELD(DST$VS HDR_FIELDS),      ! cases DST record
2371 2490 2     DSTPTR: REF DST$RECORD,        ! Pointer to DST Record
2372 2491 2     VALSPEC: REF DST$VAL_SPEC;     ! Pointer to DST record's value spec
2373 2492 2
2374 2493 2
2375 2494 2 VALSPEC = DBG$STA_SYMADDR(.SYMID, ADDRPTR, ADDRKIND);
2376 2495 2 CASE .ADDRKIND FROM DBG$K_ADDRKIND_LITERAL TO DBG$K_ADDRKIND_INVALID OF
2377 2496 2     SET
2378 2497 2
2379 2498 2         ! This is a literal value.
2380 2499 2         !
2381 2500 2         [DBG$K_ADDRKIND_LITERAL]:
2382 2501 2         BEGIN
2383 2502 2             DBG$PRINT(UPLIT BYTE(%ASCIC 'constant: !SL'), ..ADDRPTR[0]);
2384 2503 2         END;
2385 2504 2
2386 2505 2
2387 2506 2         ! This is an address or descriptor representation.
2388 2507 2         !
2389 2508 2         [DBG$K_ADDRKIND_ADDR, DBG$K_ADDRKIND_DESC]:
2390 2509 2         BEGIN
2391 2510 2             IF .ADDRKIND EQL DBG$K_ADDRKIND_ADDR
2392 2511 2             THEN
2393 2512 2                 DBG$PRINT(UPLIT BYTE(%ASCIC 'address: '), 0)
2394 2513 2             ELSE
2395 2514 2                 DBG$PRINT(UPLIT BYTE(%ASCIC 'descriptor address: '), 0);
2396 2515 2
2397 2516 2
2398 2517 2
2399 2518 2         ! For Bliss Data Items, if the address interpretation is not
2400 2519 2         ! complete, there are some extra work has to be done in order
2401 2520 2         ! to get the contents of the Val Spec. (This excersize is done
2402 2521 2         ! in DBG$STA_ADDRSPEC, unfortunately, the value needed is local to
2403 2522 2         ! that routine, so this piece of code is reproduce that value
```



```
2404 2523 3
2405 2524 3
2406 2525 3
2407 2526 4
2408 2527 3
2409 2528 4
2410 2529 4
2411 2530 4
2412 2531 4
2413 2532 4
2414 2533 4
2415 2534 4
2416 2535 4
2417 2536 4
2418 2537 4
2419 2538 4
2420 2539 4
2421 2540 4
2422 2541 4
2423 2542 3
2424 2543 3
2425 2544 4
2426 2545 4
2427 2546 4
2428 2547 4
2429 2548 4
2430 2549 4
2431 2550 4
2432 2551 5
2433 2552 5
2434 2553 5
2435 2554 5
2436 2555 5
2437 2556 5
2438 2557 5
2439 2558 5
2440 2559 5
2441 2560 5
2442 2561 5
2443 2562 5
2444 2563 5
2445 2564 5
2446 2565 5
2447 2566 5
2448 2567 5
2449 2568 4
2450 2569 4
2451 2570 4
2452 2571 4
2453 2572 4
2454 2573 4
2455 2574 4
2456 2575 4
2457 2576 4
2458 2577 3
2459 2578 3
2460 2579 3
```

```
! from the DST)
DSTPTR = .SYMID[RST$DSTPTR];
IF (.ADDRPTR[0] LSS 0) AND (.DSTPTR[DST$B_TYPE] EQL DST$K_BLI)
THEN
  BEGIN
    BLIVALSPEC[DST$B_VS_VFLAGS] = .DSTPTR[DST$B_BLI_VFLAGS];
    BLITRLR = DSTPTR[DST$A_BLI_TRLR1] + .DSTPTR[DST$B_BLI_LNG];
    BLIVALSPEC[DST$B_VS_VALUE] = .BLITRLR[DST$B_BLI_VALUE];
    VALSPEC = BLIVALSPEC[DST$B_VS_VFLAGS];
    WHILE .VALSPEC[DST$B_VS_VFLAGS] EQL DST$K_VFLAGS TVS DO
      VALSPEC = VALSPEC[DST$A_VS_TVS_BASE] + .VALSPEC[DST$B_VS_TVS_OFFSET];
    END;

! Check to see if the address interpretation is complete or not.
! If it is incomplete (indicated by -1) then we have some more
! work to do.
IF .ADDRPTR[0] LSS 0
THEN
  BEGIN
    IF .VALSPEC[DST$V_VS_INDIRECT]
    THEN
      DBG$PRINT(UPLIT BYTE(%ASCIC '()'), 0);

    IF .VALSPEC[DST$V_VS_DISP]
    THEN
      BEGIN
        LOCAL
          REG_NUM;

        DBG$PRINT(UPLIT BYTE(%ASCIC ' '), 0);
        REG_NUM = .VALSPEC[DST$V_VS_REGNUM];
        DBG$TRANSLATE REG(REG_NUM);
        IF .VALSPEC[DST$B_VS_VALUE] LSS 0
        THEN
          DBG$PRINT(UPLIT BYTE(%ASCIC '!SL'), .VALSPEC[DST$B_VS_VALUE]);

        IF .VALSPEC[DST$B_VS_VALUE] GTR 0
        THEN
          DBG$PRINT(UPLIT BYTE(%ASCIC '+!UL'), .VALSPEC[DST$B_VS_VALUE]);

        END
      ELSE
        DBG$PRINT(UPLIT BYTE(%ASCIC '!XL'), .VALSPEC[DST$B_VS_VALUE]);

    IF .VALSPEC[DST$V_VS_INDIRECT]
    THEN
      DBG$PRINT(UPLIT BYTE(%ASCIC ')'), 0);

    END
  ELSE
    DBG$PRINT(UPLIT BYTE(%ASCIC '!XL'), .ADDRPTR[0]);
```



```
: 2461      2580      2
: 2462      2581
: 2463      2582
: 2464      2583
: 2465      2584
: 2466      2585
: 2467      2586
: 2468      2587
: 2469      2588
: 2470      2589
: 2471      2590
: 2472      2591
: 2473      2592
: 2474      2593
: 2475      2594
: 2476      2595
: 2477      2596
: 2478      2597
: 2479      2598
: 2480      2599
: 2481      2600
: 2482      2601
: 2483      2602
: 2484      2603
: 2485      2604
: 2486      2605
: 2487      2606
: 2488      2607
: 2489      2608
: 2490      2609
: 2491      2610
: 2492      2611
: 2493      2612
: 2494      2613
: 2495      2614
: 2496      2615
: 2497      2616
: 2498      2617
: 2499      2618
: 2500      2619
: 2501      2620
: 2502      2621
: 2503      2622
: 2504      2623
: 2505      2624
: 2506      2625
: 2507      2626
: 2508      2627
: 2509      2628
: 2510      2629
: 2511      2630
: 2512      2631
: 2513      2632
: 2514      2633
: 2515      2634
: 2516      2635
: 2517      2636

END;

! This a register.
[DBG$K_ADDRKIND_REG]:
BEGIN
DBG$PRINT(UPLIT BYTE(%ASCIC 'address: '), 0);
DBG$TRANSLATE_REG(.ADDRPTR[0]);
END;

! This is an absolute address.
[DBG$K_ADDRKIND_ABS]:
BEGIN
DBG$PRINT(UPLIT BYTE(%ASCIC 'address: !XL'), .ADDRPTR[0]);

! The second longword may have contain the end address of the
! lexical entity. But, for global routine, start address is
! the same as end address in RST entry. And also for Entry point,
! or Label, there is no end address.
IF (.ADDRPTR[1] NEQ 0) AND (.ADDRPTR[0] NEQ .ADDRPTR[1])
THEN
DBG$PRINT(UPLIT BYTE(%ASCIC ' size: !UL bytes'),
.ADDRPTR[1] - .ADDRPTR[0] + 1);
END;

! This address is a bit offsets.
[DBG$K_ADDRKIND_BITOFF]:
BEGIN
DBG$PRINT(UPLIT BYTE
(%ASCIC 'address: offset !UL bytes'), .ADDRPTR[0]);
IF .ADDRPTR[1] NEQ 0
THEN
DBG$PRINT(UPLIT BYTE(%ASCIC ' !UL bits'), .ADDRPTR[1]);

DBG$PRINT(UPLIT BYTE(%ASCIC ' from beginning of record'), 0);
END;

! This is Bliss Field data.
[DBG$K_ADDRKIND_BLIFLD]:
BEGIN
LOCAL
FLDCOMP_PTR: REF VECTOR;

DBG$PRINT(UPLIT BYTE(%ASCIC 'bliss field:'), 0);
DSTPTR = .SYMIDCRST$L_DSTPTR;
FLDCOMP_PTR
= DSTPTR[DST$B_BLIFLD_NAME] + .DSTPTR[DST$B_BLIFLD_NAME] + 1;
DBG$PRINT(UPLIT BYTE(%ASCIC ' ['), 0);
```



```
: 2518      2637  3      INCR FLDCOMP_NO FROM 1 TO .DSTPTR[DST$L_BLI_FLD_COMPS] DO
: 2519      2638  4      BEGIN
: 2520      2639  5      IF (.FLDCOMP_NO EQL .DSTPTR[DST$L_BLI_FLD_COMPS])
: 2521      2640  4      THEN
: 2522      2641  4          DBG$PRINT(UPLIT BYTE(%ASCIC '!SL'), .FLDCOMP_PTR[FLDCOMP_NO - 1])
: 2523      2642  4      ELSE
: 2524      2643  4          DBG$PRINT(UPLIT BYTE(%ASCIC '!SL,'), .FLDCOMP_PTR[FLDCOMP_NO - 1]);
: 2525      2644  3      END;
: 2526      2645  3
: 2527      2646  2      END;
: 2528      2647  2
: 2529      2648  2
: 2530      2649  2      ! This address is too complex to calculate.
: 2531      2650  2      [DBG$K_ADDRKIND_COMPLEX]:
: 2532      2651  2      BEGIN
: 2533      2652  3      DBG$PRINT(UPLIT BYTE(%ASCIC 'address: complex address computation'), 0);
: 2534      2653  3      END;
: 2535      2654  2
: 2536      2655  2
: 2537      2656  2
: 2538      2657  2      ! Invalid address representation.
: 2539      2658  2      [DBG$K_ADDRKIND_INVALID]:
: 2540      2659  2      BEGIN
: 2541      2660  3      DBG$PRINT(UPLIT BYTE(%ASCIC 'address: invalid address'), 0);
: 2542      2661  3      END;
: 2543      2662  2
: 2544      2663  2
: 2545      2664  2      TES;
: 2546      2665  2
: 2547      2666  2      DBG$NEWLINE();
: 2548      2667  2      RETURN 0;
: 2549      2668  2
: 2550      2669  1      END;
```

```
                                .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0
      4C  53  21  20  3A  74  6E  61  74  73  6E  6F  63  0D  00896 P.AFO:  .ASCII  <13>\constant: !SL\
64  64  61  20  72  6F  74  70  69  72  63  73  65  64  14  008A4 P.AFP:  .ASCII  <9>\address: \
      20  3A  73  73  65  72  64  61  09  008AE P.AFQ:  .ASCII  <20>\descriptor address: \
      20  3A  73  73  65  72  008BD
      28  2E  02  008C3 P.AFR:  .ASCII  <2>\.( \
      2E  01  008C6 P.AFS:  .ASCII  <1>\.( \
      4C  53  21  2B  04  008C8 P.AFT:  .ASCII  <3>\!SL\
      4C  58  21  03  008CC P.AFU:  .ASCII  <4>\+!UL\
      29  01  008D1 P.AFV:  .ASCII  <3>\!XL\
      21  03  008D5 P.AFW:  .ASCII  <1>\)\
      4C  58  21  03  008D7 P.AFX:  .ASCII  <3>\!XL\
      64  64  61  09  008DB P.AFY:  .ASCII  <9>\address: \
79  62  4C  58  21  20  3A  73  73  65  72  64  64  61  0C  008E5 P.AFZ:  .ASCII  <12>\address: !XL\
      20  2C  11  008F2 P.AGA:  .ASCII  <17>\, size: !UL bytes\
      73  65  74  00901
65  73  66  66  6F  20  3A  73  73  65  72  64  64  61  19  00904 P.AGB:  .ASCII  <25>\address: offset !UL bytes\
      73  65  74  79  62  20  4C  55  21  20  74  00913
      73  74  69  62  20  4C  55  21  20  09  0091E P.AGC:  .ASCII  <9>\ !UL bits\
6E  69  6E  6E  69  67  65  62  20  6D  6F  72  66  20  19  00928 P.AGD:  .ASCII  <25>\ from beginning of record\
      72  66  20  19
```



|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |  |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|--|
| 6C | 70 | 6D | 6F | 63 | 20 | 3A | 73 | 73 | 65 | 72 | 64 | 61 | 74 | 75 | 0097A |  |
| 70 | 6D | 6F | 63 | 20 | 73 | 73 | 65 | 72 | 64 | 64 | 61 | 20 | 78 | 65 | 0096B |  |
| 6C | 61 | 76 | 6E | 69 | 20 | 3A | 73 | 73 | 65 | 72 | 64 | 61 | 74 | 75 | 0097A |  |
|    |    |    |    |    | 73 | 73 | 65 | 72 | 64 | 64 | 61 | 20 | 64 | 69 | 00990 |  |

P.AGE: .ASCII <12>\bliss field:\  
P.AGF: .ASCII <2>\ [\   
P.AGG: .ASCII <4>\!SL]\   
P.AGH: .ASCII <4>\!SL,\   
P.AGI: .ASCII \\$address: complex address computation\  
P.AGJ: .ASCII <24>\address: invalid address\

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

007C 00000 SHOW\_SYMBOL ADDRESS:

|      |           |      |      |       |        |                      |                |                        |      |
|------|-----------|------|------|-------|--------|----------------------|----------------|------------------------|------|
| 56   | 00000000G | 00   | 9E   | 00002 | .WORD  | Save R2,R3,R4,R5,R6  | 2466           |                        |      |
| 55   | 00000000' | EF   | 9E   | 00009 | MOVAB  | DBG\$PRINT, R6       |                |                        |      |
| 5E   |           | 14   | C2   | 00010 | MOVAB  | P.AFO, R5            |                |                        |      |
|      |           | 5E   | DD   | 00013 | SUBL2  | #20, SP              |                |                        |      |
|      |           | 10   | AE   | 9F    | PUSHL  | SP                   | 2494           |                        |      |
| 53   |           | 04   | AC   | DD    | PUSHAB | ADDRPTR              |                |                        |      |
|      |           | 53   | DD   | 00018 | MOVL   | SYMPID, R3           |                |                        |      |
| F231 | CF        | 03   | FB   | 0001E | PUSHL  | R3                   |                |                        |      |
|      | 54        | 50   | DD   | 00023 | CALLS  | #3, DBG\$STA_SYMADDR |                |                        |      |
|      | 00        | 6E   | CF   | 00026 | MOVL   | R0, VALSPEC          |                |                        |      |
| 00C3 | 08        |      |      |       | CASEL  | ADDRKIND, #0, #8     | 2495           |                        |      |
| 015C | 001A      | 001A | 0012 | 0002A | .WORD  | 2\$-1\$,-            |                |                        |      |
|      | 0117      | 00F7 | 00D6 | 00032 |        | 3\$-1\$,-            |                |                        |      |
|      |           |      | 0164 | 0003A |        | 3\$-1\$,-            |                |                        |      |
|      |           |      |      |       |        | 14\$-1\$,-           |                |                        |      |
|      |           |      |      |       |        | 16\$-1\$,-           |                |                        |      |
|      |           |      |      |       |        | 17\$-1\$,-           |                |                        |      |
|      |           |      |      |       |        | 20\$-1\$,-           |                |                        |      |
|      |           |      |      |       |        | 25\$-1\$,-           |                |                        |      |
|      |           |      |      |       |        | 26\$-1\$             |                |                        |      |
|      |           |      | 0C   | BE    | DD     | 0003C                | 2\$: PUSHAB    | 2503                   |      |
|      |           |      | 55   | DD    | 0003F  |                      | PUSHL          |                        |      |
|      |           |      | 0150 | 31    | 00041  | BRW                  | 27\$           |                        |      |
|      |           | 01   | 6E   | D1    | 00044  | 3\$: CMPL            | ADDRKIND, #1   | 2511                   |      |
|      |           |      | 07   | 12    | 00047  | BNEQ                 | 4\$            |                        |      |
|      |           |      | 7E   | D4    | 00049  | CLRL                 | -(SP)          | 2513                   |      |
|      |           | 0E   | A5   | 9F    | 0004B  | PUSHAB               | P.AFP          |                        |      |
|      |           |      | 05   | 11    | 0004E  | BRB                  | 5\$            |                        |      |
|      |           |      | 7E   | D4    | 00050  | 4\$: CLRL            | -(SP)          | 2515                   |      |
|      |           | 18   | A5   | 9F    | 00052  | PUSHAB               | P.AFO          |                        |      |
|      |           | 66   | 02   | FB    | 00055  | 5\$: CALLS           | #2, DBG\$PRINT |                        |      |
|      |           | 52   | 0C   | A3    | 00058  | MOVL                 | 12(R3), DSTPTR | 2525                   |      |
|      |           |      | 0C   | 51    | 0005C  | CLRL                 | R1             | 2526                   |      |
|      |           |      | 0C   | AE    | 0005E  | TSTL                 | ADDRPTR        |                        |      |
|      |           |      | 2E   | 18    | 00061  | BGEQ                 | 7\$            |                        |      |
|      |           |      | 51   | D6    | 00063  | INCL                 | R1             |                        |      |
|      |           | 01   | A2   | 95    | 00065  | TSTB                 | 1(DSTPTR)      |                        |      |
|      |           |      | 27   | 12    | 00068  | BNEQ                 | 7\$            |                        |      |
|      | 04        | AE   | 04   | A2    | 90     | 0006A                | MOVAB          | 4(DSTPTR), BLIVALSPEC  | 2529 |
|      |           | 50   | 02   | A2    | 9A     | 0006F                | MOVZBL         | 2(DSTPTR), R0          | 2530 |
|      |           | 50   | 03   | A042  | 9E     | 00073                | MOVAB          | 3(R0)[DSTPTR], BLITRLR |      |



|    |      |    |    |      |    |       |        |                            |      |
|----|------|----|----|------|----|-------|--------|----------------------------|------|
|    | 05   | AE |    | 60   | D0 | 00078 | MOVL   | (BLITRLR), BLIVALSPEC+1    | 2531 |
|    |      | 54 |    | AE   | 9E | 0007C | MOVAB  | BLIVALSPEC, VALSPEC        | 2532 |
|    | FB   | 8F |    | 64   | 91 | 00080 | CMPB   | (VALSPEC), #251            | 2533 |
|    |      |    |    | 0B   | 12 | 00084 | BNEQ   | 7\$                        |      |
| 50 |      | 54 |    | A4   | C1 | 00086 | ADDL3  | 1(VALSPEC), VALSPEC, R0    | 2534 |
|    |      | 54 |    | A0   | 9E | 0008B | MOVAB  | 5(R0), VALSPEC             |      |
|    |      |    |    | EF   | 11 | 0008F | BRB    | 6\$                        |      |
|    |      | 51 |    | 51   | E9 | 00091 | 7\$:   | R1, 13\$                   | 2542 |
| 08 |      | 64 |    | 02   | E1 | 00094 | BLBC   | #2, (VALSPEC), 8\$         | 2545 |
|    |      |    |    | 7E   | D4 | 00098 | BBC    | -(SP)                      | 2547 |
|    |      |    | 2D | A5   | 9F | 0009A | CLRL   | P.AFR                      |      |
|    |      | 66 |    | 02   | FB | 0009D | PUSHAB | #2, DBG\$PRINT             |      |
| 2D |      | 64 |    | 03   | E1 | 000A0 | CALLS  | #2, (VALSPEC), 10\$        | 2549 |
|    |      |    |    | 7E   | D4 | 000A4 | BBC    | -(SP)                      | 2555 |
|    |      |    | 30 | A5   | 9F | 000A6 | CLRL   | P.AFS                      |      |
|    |      | 66 |    | 02   | FB | 000A9 | PUSHAB | #2, DBG\$PRINT             |      |
| 50 | 64   | 04 |    | 04   | EF | 000AC | CALLS  | #4, #4, (VALSPEC), REG_NUM | 2556 |
|    |      |    |    | 50   | DD | 000B1 | EXTZV  | REG_NUM                    | 2557 |
|    | F8AE | CF |    | 01   | FB | 000B3 | PUSHL  | REG_NUM                    |      |
|    |      | 52 |    | 01   | A4 | D0    | CALLS  | #1, DBG\$TRANSALATE_REG    | 2558 |
|    |      |    |    | 08   | D0 | 000B8 | MOVL   | 1(VALSPEC), R2             |      |
|    |      |    |    | 52   | 18 | 000BC | BGEQ   | 9\$                        |      |
|    |      |    | 32 | A5   | 9F | 000C0 | PUSHL  | R2                         | 2560 |
|    |      | 66 |    | 02   | FB | 000C3 | PUSHAB | P.AFT                      |      |
|    |      |    |    | 52   | D5 | 000C6 | CALLS  | #2, DBG\$PRINT             |      |
|    |      |    |    | 10   | 15 | 000C8 | TSTL   | R2                         | 2562 |
|    |      |    |    | 52   | DD | 000CA | BLEQ   | 12\$                       |      |
|    |      |    | 36 | A5   | 9F | 000CC | PUSHL  | R2                         | 2564 |
|    |      |    |    | 06   | 11 | 000CF | PUSHAB | P.AFU                      |      |
|    |      |    |    | 01   | A4 | DD    | BRB    | 11\$                       |      |
|    |      |    |    | 3B   | A5 | 9F    | PUSHL  | 1(VALSPEC)                 | 2569 |
|    |      | 66 |    | 02   | FB | 000D4 | PUSHAB | P.AFV                      |      |
| 1F |      | 64 |    | 02   | E1 | 000D7 | CALLS  | #2, DBG\$PRINT             |      |
|    |      |    |    | 02   | E1 | 000DA | BBC    | #2, (VALSPEC), 15\$        | 2571 |
|    |      |    |    | 7E   | D4 | 000DE | CLRL   | -(SP)                      | 2573 |
|    |      |    | 3F | A5   | 9F | 000E0 | PUSHL  | P.AFW                      |      |
|    |      |    |    | 5A   | 11 | 000E3 | PUSHAB | 19\$                       |      |
|    |      |    |    | 0C   | AE | DD    | BRB    | 19\$                       |      |
|    |      |    |    | 41   | A5 | 9F    | PUSHL  | ADDRPTR                    | 2578 |
|    |      |    |    | 52   | 11 | 000E8 | PUSHAB | P.AFX                      |      |
|    |      |    |    | 7E   | D4 | 000ED | BRB    | 19\$                       |      |
|    |      |    | 45 | A5   | 9F | 000EF | CLRL   | -(SP)                      | 2587 |
|    |      | 66 |    | 02   | FB | 000F2 | PUSHAB | P.AFY                      |      |
|    |      |    |    | 0C   | AE | DD    | CALLS  | #2, DBG\$PRINT             |      |
|    | F869 | CF |    | 01   | FB | 000F8 | PUSHL  | ADDRPTR                    | 2588 |
|    |      |    |    | 0097 | 31 | 000FD | CALLS  | #1, DBG\$TRANSALATE_REG    |      |
|    |      |    |    | 0C   | AE | DD    | BRW    | 28\$                       | 2495 |
|    |      |    |    | 4F   | A5 | 9F    | PUSHL  | ADDRPTR                    | 2596 |
|    |      | 66 |    | 02   | FB | 00103 | PUSHAB | P.AFZ                      |      |
|    |      | 50 |    | 10   | AE | D0    | CALLS  | #2, DBG\$PRINT             |      |
|    |      |    |    | EE   | 13 | 0010D | MOVL   | ADDRPTR+4, R0              | 2604 |
|    |      | 50 |    | 0C   | AE | D1    | BEQL   | 15\$                       |      |
|    |      |    |    | E8   | 13 | 0010F | CMPL   | ADDRPTR, R0                |      |
|    |      |    |    | 0C   | AE | C2    | BEQL   | 15\$                       |      |
|    |      | 50 |    | 01   | A0 | 9F    | SUBL2  | ADDRPTR, R0                | 2607 |
|    |      |    |    | 5C   | A5 | 9F    | PUSHAB | 1(R0)                      |      |
|    |      |    |    | 73   | 11 | 0011F | PUSHAB | P.AGA                      | 2606 |
|    |      |    |    | 0C   | AE | DD    | BRB    | 27\$                       |      |
|    |      |    |    |      |    |       | PUSHL  | ADDRPTR                    | 2616 |



|           |      |      |       |       |        |                             |      |
|-----------|------|------|-------|-------|--------|-----------------------------|------|
| 66        | 6E   | A5   | 9F    | 00124 | PUSHAB | P,AGB                       | 2615 |
|           |      | 02   | FB    | 00127 | CALLS  | #2, DBG\$PRINT              |      |
|           | 10   | AE   | D5    | 0012A | TSTL   | ADDRPTR+4                   | 2617 |
|           |      | 0A   | 13    | 0012D | BEQL   | 18\$                        |      |
|           | 10   | AE   | DD    | 0012F | PUSHL  | ADDRPTR+4                   | 2619 |
|           | 0088 | C5   | 9F    | 00132 | PUSHAB | P,AGC                       |      |
| 66        |      | 02   | FB    | 00136 | CALLS  | #2, DBG\$PRINT              |      |
|           |      | 7E   | D4    | 00139 | CLRL   | -(SP)                       | 2621 |
|           | 0092 | C5   | 9F    | 0013B | PUSHAB | P,AGD                       |      |
|           |      | 53   | 11    | 0013F | BRB    | 27\$                        |      |
|           |      | 7E   | D4    | 00141 | CLRL   | -(SP)                       | 2632 |
|           | 00AC | C5   | 9F    | 00143 | PUSHAB | P,AGE                       |      |
| 66        |      | 02   | FB    | 00147 | CALLS  | #2, DBG\$PRINT              |      |
| 52        | 0C   | A3   | D0    | 0014A | MOVL   | 12(R3), DSTPTR              | 2633 |
| 50        | 07   | A2   | 9A    | 0014E | MOVZBL | 7(DSTPTR), R0               | 2635 |
| 54        | 08   | A042 | 9E    | 00152 | MOVAB  | 8(R0)[DSTPTR], FLDCOMP_PTR  |      |
|           |      | 7E   | D4    | 00157 | CLRL   | -(SP)                       | 2636 |
|           | 00B9 | C5   | 9F    | 00159 | PUSHAB | P,AGF                       |      |
| 66        |      | 02   | FB    | 0015D | CALLS  | #2, DBG\$PRINT              |      |
|           |      | 53   | D4    | 00160 | CLRL   | FLDCOMP_NO                  | 2641 |
|           |      | 1B   | 11    | 00162 | BRB    | 24\$                        |      |
| 03        | A2   | 53   | D1    | 00164 | CMPL   | FLDCOMP_NO, 3(DSTPTR)       | 2639 |
|           |      | 0A   | 12    | 00168 | BNEQ   | 22\$                        |      |
|           | FC   | A443 | DD    | 0016A | PUSHL  | -4(FLDCOMP_PTR)[FLDCOMP_NO] | 2641 |
|           | 00BC | C5   | 9F    | 0016E | PUSHAB | P,AGG                       |      |
|           |      | 08   | 11    | 00172 | BRB    | 23\$                        |      |
|           | FC   | A443 | DD    | 00174 | PUSHL  | -4(FLDCOMP_PTR)[FLDCOMP_NO] | 2643 |
|           | 00C1 | C5   | 9F    | 00178 | PUSHAB | P,AGH                       |      |
| 66        |      | 02   | FB    | 0017C | CALLS  | #2, DBG\$PRINT              |      |
| E0        | 53   | 03   | A2    | F3    | AOBLEQ | 3(DSTPTR), FLDCOMP_NO, 21\$ | 2637 |
|           |      | 11   | 11    | 00184 | BRB    | 28\$                        | 2495 |
|           |      | 7E   | D4    | 00186 | CLRL   | -(SP)                       | 2653 |
|           | 00C6 | C5   | 9F    | 00188 | PUSHAB | P,AGI                       |      |
|           |      | 06   | 11    | 0018C | BRB    | 27\$                        |      |
|           |      | 7E   | D4    | 0018E | CLRL   | -(SP)                       | 2661 |
|           | 00EB | C5   | 9F    | 00190 | PUSHAB | P,AGJ                       |      |
|           |      | 02   | FB    | 00194 | CALLS  | #2, DBG\$PRINT              |      |
| 00000000G | 66   | 00   | FB    | 00197 | CALLS  | #0, DBG\$NEWLINE            | 2666 |
|           |      | 04   | 0019E | RET   |        |                             | 2669 |

; Routine Size: 415 bytes, Routine Base: DBG\$CODE + 1157



```
2552 2670 1 ROUTINE SHOW_SYMBOL_TYPE(SYMID): NOVALUE =
2553 2671 1
2554 2672 1 FUNCTION
2555 2673 1     This routine takes a SYMID and shows the type information associated
2556 2674 1     with the input SYMID. This is a recursive routine.
2557 2675 1
2558 2676 1 INPUTS
2559 2677 1     SYMID - The SYMID for the symbol.
2560 2678 1
2561 2679 1 OUTPUTS
2562 2680 1     None.
2563 2681 1
2564 2682 1
2565 2683 2 BEGIN
2566 2684 2
2567 2685 2 MAP
2568 2686 2     SYMID: REF RST$ENTRY;           ! Pointer to input symbol's RST entry
2569 2687 2
2570 2688 2 ENABLE
2571 2689 2     SHOW_SYMBOL_TYPE_HANDLER;     ! Set up error handler for this routine
2572 2690 2
2573 2691 2 LOCAL
2574 2692 2     AREA_ADDR,                   ! Byte address of the Area associated
2575 2693 2                                     ! with the given offset
2576 2694 2     AREA_LEN,                     ! Byte length of the Area associated
2577 2695 2                                     ! with the given offset
2578 2696 2     BITSIZE,                     ! The size in bits of an item of the
2579 2697 2                                     ! given type
2580 2698 2     CELLTYPE: REF RST$ENTRY,       ! The Type ID of the individual array
2581 2699 2                                     ! element's data type
2582 2700 2     DSCADDR: REF BLOCK[,BYTE],     ! VAX Standard Descriptor
2583 2701 2     DSTPTR: REF DST$RECORD,        ! Pointer to symbol's DST record
2584 2702 2     ELTVECPTR: REF VECTOR[,LONG], ! Pointer to a vector of SYMIDs for the
2585 2703 2                                     ! enumeration type elements, or
2586 2704 2                                     ! record components, or
2587 2705 2                                     ! variant components, or
2588 2706 2                                     ! a subscript vector contains
2589 2707 2                                     ! subscript Type ID, one longword
2590 2708 2                                     ! per dimension
2591 2709 2     FCODE,                       ! Symbol's FCODE field
2592 2710 2     KIND,                       ! Symbol's KIND field
2593 2711 2     HIGHPTR,                    ! Upper bound value of the subrange
2594 2712 2     LANGCODE,                   ! Language Code
2595 2713 2     LANGUAGE: REF VECTOR[,BYTE], ! Pointer to counted ASCII string
2596 2714 2     LOWPTR,                     ! Lower bound value of the subrange
2597 2715 2     NAME: REF VECTOR[,BYTE],     ! Counted ASCII String
2598 2716 2     NELTS,                      ! The number of enumeration type elements
2599 2717 2                                     ! or record components,
2600 2718 2                                     ! or variant components,
2601 2719 2                                     ! or array dimensions
2602 2720 2     PARENT_TYPE: REF RST$ENTRY,   ! The Type ID of the set's parent type
2603 2721 2     PICT_DESC : BLOCK [8,BYTE],   ! VAX Standard Descriptor
2604 2722 2     PICT_LENGTH,                 ! The length of the picture representation
2605 2723 2     PICT_LENPTR: REF VECTOR[,BYTE], ! Pointer to the picture representation
2606 2724 2     PICTPTR: REF VECTOR[,BYTE],   ! Pointer to the picture representation
2607 2725 2     PICTVAL,                    ! Pointer to the language-specific
2608 2726 2                                     ! encoding
```



```
: 2609      2727      2      POOLID,      : Index to Temporary Memory Pool Stack
: 2610      2728      2      PSCALE,      : Scale factor and digit count for the
: 2611      2729      2      : picture item
: 2612      2730      2      REFTYPID: REF RST$ENTRY,      : Type ID for target type
: 2613      2731      2      TYPECODE,      : Type code
: 2614      2732      2      TYPEID: REF RST$ENTRY;      : Symbol's TYPE RST entry
: 2615      2733      2
: 2616      2734      2
: 2617      2735      2      POOLID = DBG$PUSH_TEMP_MEM();
: 2618      2736      2      DBG$STA_SYMTYPE(.SYMID, FCODE, TYPEID);
: 2619      2737      2      CASE .FCODE FROM RST$K_TYPE_MINIMUM TO RST$K_TYPE_MAXIMUM OF
: 2620      2738      2      SET
: 2621      2739      2
: 2622      2740      2      [RST$K_TYPE_ARRAY]:
: 2623      2741      2      BEGIN
: 2624      2742      2      DBG$STA_TYP_ARRAY(.TYPEID, DSCADDR, CELLTYPE, NELTS, ELTVEC_PTR, BITSIZE);
: 2625      2743      2      DBG$TRANSLATE_DESCR(.DSCADDR);
: 2626      2744      2      ITEM_SIZE(BITSIZE);
: 2627      2745      2      DBG$NEWLINE();
: 2628      2746      2      DBG$PRINT_CONTROL(DBG$K_PRTSET_RLMARGIN, +4);
: 2629      2747      2      DBG$PRINT(UP_LIT_BYTE(%ASCIC 'cell type: '), 0);
: 2630      2748      2      DBG$PRINT_FLAG = TRUE;
: 2631      2749      2      SHOW_SYMBOL_TYPE(.CELLTYPE);
: 2632      2750      2      DBG$PRINT_CONTROL(DBG$K_PRTSET_RLMARGIN, -4);
: 2633      2751      2      END;
: 2634      2752      2
: 2635      2753      2      [RST$K_TYPE_ATOMIC]:
: 2636      2754      2      BEGIN
: 2637      2755      2      DBG$PRINT(UP_LIT_BYTE(%ASCIC 'atomic type: '), 0);
: 2638      2756      2      DBG$STA_TYP_ATOMIC(.TYPEID, TYPECODE, BITSIZE);
: 2639      2757      2      DBG$TRANSLATE_TYPECODE(.TYPECODE);
: 2640      2758      2      ITEM_SIZE(BITSIZE);
: 2641      2759      2      END;
: 2642      2760      2
: 2643      2761      2      [RST$K_TYPE_DESCR]:
: 2644      2762      2      BEGIN
: 2645      2763      2      DBG$STA_TYP_DESCR(.TYPEID, DSCADDR);
: 2646      2764      2      DBG$TRANSLATE_DESCR(.DSCADDR);
: 2647      2765      2      END;
: 2648      2766      2
: 2649      2767      2      [RST$K_TYPE_ENUM]:
: 2650      2768      2      BEGIN
: 2651      2769      2      DBG$PRINT(UP_LIT_BYTE(%ASCIC 'enumeration type'), 0);
: 2652      2770      2      DBG$STA_TYP_ENUM(.TYPEID, NELTS, ELTVEC_PTR, BITSIZE);
: 2653      2771      2      NAME = DBG$GET_DST_NAME(.TYPEID[RST$L_DST_PTR]);
: 2654      2772      2      IF .NAME[0] NEQ 0
: 2655      2773      2      THEN
: 2656      2774      2      DBG$PRINT(UP_LIT_BYTE(%ASCIC ' (!AC,'). .NAME)
: 2657      2775      2      ELSE
: 2658      2776      2      DBG$PRINT(UP_LIT_BYTE(%ASCIC ' (anonymous,'). 0);
: 2659      2777      2
: 2660      2778      2      DBG$PRINT(UP_LIT_BYTE(%ASCIC ' !UL elements)'), .NELTS);
: 2661      2779      2      ITEM_SIZE(BITSIZE);
: 2662      2780      2      END;
: 2663      2781      2
: 2664      2782      2      [RST$K_TYPE_PICT]:
: 2665      2783      2      BEGIN
```



```
: 2666      2784      3      DBG$PRINT(UPLIT BYTE(%ASCIC 'picture type'), 0);
: 2667      2785      3      DBG$STA TYP PICT(.TYPEID, LANGCODE, PICTPTR, PICTVAL, PSscale);
: 2668      2786      3      LANGUAGE = DBG$LANGUAGE(.LANGCODE);
: 2669      2787      3      DBG$PRINT(UPLIT BYTE(%ASCIC ' !AC'), .LANGUAGE);
: 2670      2788      3      IF .PICTPTR[0] NEQ 0
: 2671      2789      3      THEN
: 2672      2790      4      BEGIN
: 2673      2791      4      INIT_DESCRIPTOR(PICT_DESC);
: 2674      2792      4      PICT_LENPTR = .PICTPTR - 1;
: 2675      2793      4      PICT_LENGTH = .PICT_LENPTR[0];
: 2676      2794      4      INCR PICT_INDEX FROM 1 TO (.PICT_LENGTH/2) DO
: 2677      2795      5      BEGIN
: 2678      2796      5      LOCAL
: 2679      2797      5      DUPL_STATUS,
: 2680      2798      5      CAT_STATUS,
: 2681      2799      5      DUPL_DESC : BLOCK [8, BYTE];
: 2682      2800      5
: 2683      2801      5      INIT_DESCRIPTOR(DUPL_DESC);
: 2684      2802      5      DUPL_STATUS = STR$DUPL_CHAR(DUPL_DESC,
: 2685      2803      5      %REF(.PICTPTR[(.PICT_INDEX - 1)*2] + 1)),
: 2686      2804      5      %REF(.PICTPTR[(.PICT_INDEX - 1)*2] + 0));
: 2687      2805      5      CAT_STATUS = STR$CONCAT(PICT_DESC, PICT_DESC, DUPL_DESC);
: 2688      2806      5      DISCARD_DESCRIPTOR(DUPL_DESC);
: 2689      2807      4      END;
: 2690      2808      4
: 2691      2809      4      DBG$PRINT(UPLIT BYTE(%ASCIC ' pic: !AS'), PICT_DESC);
: 2692      2810      4      DISCARD_DESCRIPTOR(PICT_DESC);
: 2693      2811      3      END;
: 2694      2812      3
: 2695      2813      3      END;
: 2696      2814      3
: 2697      2815      3      [RST$K_TYPE_IPTR]:
: 2698      2816      3      BEGIN
: 2699      2817      3      DBG$PRINT(UPLIT BYTE(%ASCIC 'typed pointer type'), 0);
: 2700      2818      3      DBG$NEWLINE();
: 2701      2819      3      DBG$STA TYP TYPEDPTR(.TYPEID, REFTYPID);
: 2702      2820      3      DBG$PRINT CONTROL(DBG$K_PRTSET_RLMARGIN, +4);
: 2703      2821      3      DBG$PRINT(UPLIT BYTE(%ASCIC 'target type: '), 0);
: 2704      2822      3      DBG$PRINT FLAG = TRUE;
: 2705      2823      3      SHOW_SYMBOL TYPE(.REFTYPID);
: 2706      2824      3      DBG$PRINT CONTROL(DBG$K_PRTSET_RLMARGIN, -4);
: 2707      2825      3      END;
: 2708      2826      3
: 2709      2827      3      [RST$K_TYPE_RECORD]:
: 2710      2828      3      BEGIN
: 2711      2829      3      DBG$PRINT(UPLIT BYTE(%ASCIC 'record type'), 0);
: 2712      2830      3
: 2713      2831      3      ! If the flag indicates that no value is specified, i.e., the
: 2714      2832      3      ! object being described is a type. Check to see if this type
: 2715      2833      3      ! has a name.
: 2716      2834      3
: 2717      2835      3
: 2718      2836      3      DSTPTR = .TYPEID[RST$L_DSTPTR];
: 2719      2837      3      IF .DSTPTR[DST$B_VFLAGS] EQL DST$K_VFLAGS_NOVAL
: 2720      2838      3      THEN
: 2721      2839      4      BEGIN
: 2722      2840      4      NAME = DBG$GET_DST_NAME(.TYPEID[RST$L_DSTPTR]);
```



```
: 2723      2841      4      IF .NAME[0] NEQ 0
: 2724      2842      4      THEN
: 2725      2843      4          DBG$PRINT(UPLIT BYTE(%ASCIC ' (!AC,'), .NAME)
: 2726      2844      4      ELSE
: 2727      2845      4          DBG$PRINT(UPLIT BYTE(%ASCIC ' (anonymous,'), 0);
: 2728      2846      4      END
: 2729      2847      4
: 2730      2848      4
: 2731      2849      3      ELSE
: 2732      2850      3          DBG$PRINT(UPLIT BYTE(%ASCIC ' (anonymous,'), 0);
: 2733      2851      3
: 2734      2852      3      DBG$STA TYP RECORD(.TYPEID, NELTS, ELTVECPTR, BITSIZE);
: 2735      2853      3      DBG$PRINT(UPLIT BYTE(%ASCIC ' !UL components'), .NELTS);
: 2736      2854      3      ITEM_SIZE(BITSIZE);
: 2737      2855      3      INCR I FROM 0 TO .NELTS - 1 DO
: 2738      2856      4          BEGIN
: 2739      2857      4              DBG$STA SYMKIND(.ELTVECPTR[I], KIND);
: 2740      2858      4              IF .KIND EQL RST$K_VARIANT THEN GET_VARIANT(.ELTVECPTR[I]);
: 2741      2859      4          END;
: 2742      2860      3
: 2743      2861      2      END;
: 2744      2862      2
: 2745      2863      2      [RST$K_TYPE_SET]:
: 2746      2864      2          BEGIN
: 2747      2865      2              DBG$PRINT(UPLIT BYTE(%ASCIC 'set type'), 0);
: 2748      2866      2              DBG$STA TYP SET(.TYPEID, PARENT_TYPE, BITSIZE);
: 2749      2867      2              ITEM_SIZE(BITSIZE);
: 2750      2868      2              DBG$NEWLINE();
: 2751      2869      2              DBG$PRINT CONTROL(DBG$K_PRTSET_RLMARGIN, +4);
: 2752      2870      2              DBG$PRINT(UPLIT BYTE(%ASCIC 'parent type: '), 0);
: 2753      2871      2              DBG$PRINT FLAG = TRUE;
: 2754      2872      2              SHOW_SYMBOL TYPE(.PARENT_TYPE);
: 2755      2873      2              DBG$PRINT_CONTROL(DBG$K_PRTSET_RLMARGIN, -4);
: 2756      2874      2          END;
: 2757      2875      2
: 2758      2876      2      [RST$K_TYPE_SUBRNG]:
: 2759      2877      2          BEGIN
: 2760      2878      2              DBG$PRINT(UPLIT BYTE(%ASCIC 'subrange type'), 0);
: 2761      2879      2              DBG$STA TYP SUBRNG(.TYPEID, PARENT_TYPE, LOWPTR, HIGHPTR, BITSIZE);
: 2762      2880      2              ITEM_SIZE(BITSIZE);
: 2763      2881      2              DBG$PRINT(UPLIT BYTE(%ASCIC ', range:'), 0);
: 2764      2882      2              DBG$PRINT(UPLIT BYTE(%ASCIC ' !SL'), ..LOWPTR);
: 2765      2883      2              DBG$PRINT(UPLIT BYTE(%ASCIC ' !SL'), ..HIGHPTR);
: 2766      2884      2              DBG$NEWLINE();
: 2767      2885      2              DBG$PRINT CONTROL(DBG$K_PRTSET_RLMARGIN, +4);
: 2768      2886      2              DBG$PRINT(UPLIT BYTE(%ASCIC 'parent type: '), 0);
: 2769      2887      2              DBG$PRINT FLAG = TRUE;
: 2770      2888      2              SHOW_SYMBOL TYPE(.PARENT_TYPE);
: 2771      2889      2              DBG$PRINT_CONTROL(DBG$K_PRTSET_RLMARGIN, -4);
: 2772      2890      2          END;
: 2773      2891      2
: 2774      2892      2      [RST$K_TYPE_COBHACK]:
: 2775      2893      2          BEGIN
: 2776      2894      2              DBG$PRINT(UPLIT BYTE(%ASCIC ' cobol formal parameter type'), 0);
: 2777      2895      2          END;
: 2778      2896      2
: 2779      2897      2      [RST$K_TYPE_BLIDATA]:
```



```

2780      2898      3      BEGIN
2781      2899      DBGS$PRINT(UPLIT BYTE(%ASCIC 'bliss data item type'), 0);
2782      2900      DSTPTR = .SYMID[RST$S DSTPTR];
2783      2901      GET_BLISS_SPECIAL_CASES(.DSTPTR);
2784      2902      END;
2785      2903
2786      2904      [RST$K_TYPE_BLIFLD]:
2787      2905      BEGIN
2788      2906      DBGS$PRINT(UPLIT BYTE(%ASCIC 'bliss field type'), 0);
2789      2907      END;
2790      2908
2791      2909      [RST$K_TYPE_FILE]:
2792      2910      BEGIN
2793      2911      DBGS$PRINT(UPLIT BYTE(%ASCIC 'file data type,'), 0);
2794      2912      DBGS$STA TYP FILE(.TYPEID, LANGCODE, REFTYPID);
2795      2913      LANGUAGE = DBGS$LANGUAGE(.LANGCODE);
2796      2914      DBGS$PRINT(UPLIT BYTE(%ASCIC ' !AC'), .LANGUAGE);
2797      2915      IF .REFTYPID NEQ 0
2798      2916      THEN
2799      2917      BEGIN
2800      2918      DBGS$NEWLINE();
2801      2919      DBGS$PRINT CONTROL(DBGS$K_PRTSET_RLMARGIN, +4);
2802      2920      DBGS$PRINT(UPLIT BYTE(%ASCIC 'target type: '), 0);
2803      2921      DBGS$PRINT FLAG = TRUE;
2804      2922      SHOW_SYMBOL TYPE(.REFTYPID);
2805      2923      DBGS$PRINT CONTROL(DBGS$K_PRTSET_RLMARGIN, -4);
2806      2924      END;
2807      2925
2808      2926      END;
2809      2927
2810      2928      [RST$K_TYPE_PTR]:
2811      2929      BEGIN
2812      2930      DBGS$PRINT(UPLIT BYTE(%ASCIC 'untyped pointer type'), 0);
2813      2931      END;
2814      2932
2815      2933      [RST$K_TYPE_RFA]:
2816      2934      BEGIN
2817      2935      DBGS$PRINT(UPLIT BYTE(%ASCIC 'record file address'), 0);
2818      2936      END;
2819      2937
2820      2938      [RST$K_TYPE_SELF_REL_LAB]:
2821      2939      BEGIN
2822      2940      DBGS$PRINT(UPLIT BYTE(%ASCIC 'self relative label'), 0);
2823      2941      END;
2824      2942
2825      2943      [RST$K_TYPE_AREA]:
2826      2944      BEGIN
2827      2945      DBGS$PRINT(UPLIT BYTE(%ASCIC 'area type,'), 0);
2828      2946      DBGS$STA TYP AREA(.TYPEID, AREA_LEN);
2829      2947      DBGS$PRINT(UPLIT BYTE(%ASCIC ' size: !UL bytes'), .AREA_LEN);
2830      2948      END;
2831      2949
2832      2950      [RST$K_TYPE_OFFSET]:
2833      2951      BEGIN
2834      2952      DBGS$PRINT(UPLIT BYTE(%ASCIC 'offset type,'), 0);
2835      2953      DBGS$STA TYP OFFSET(.TYPEID, AREA_ADDR, AREA_LEN);
2836      2954      DBGS$PRINT(UPLIT BYTE(%ASCIC ' byte address: !XL, '), .AREA_ADDR);
```



```

2837      2955      3      DBG$PRINT(UPLIT BYTE(%ASCIC ' size: !UL bytes'), .AREA_LEN);
2838      2956      END;
2839      2957
2840      2958      [INRANGE, OUTRANGE]:
2841      2959      BEGIN
2842      2960      DBG$PRINT(UPLIT BYTE(%ASCIC 'invalid type'), 0);
2843      2961      END;
2844      2962
2845      2963      TES;
2846      2964
2847      2965      IF .DBG$_PRINT_FLAG
2848      2966      THEN
2849      2967      BEGIN
2850      2968      DBG$NEWLINE();
2851      2969      DBG$_PRINT_FLAG = FALSE;
2852      2970      END;
2853      2971
2854      2972
2855      2973      ! Release the TYPEID RST entry for this SYMID.
2856      2974      !
2857      2975      DBG$RST_TEMP RELEASE();
2858      2976      DBG$POP_TEMP MEM(.POOLID);
2859      2977      RETURN 0;
2860      2978
2861      2979      1      END;

```

|    |    |    |    |    |    |    |    |    |    |    |    |    |    |    | .PSECT | DBG\$PLIT,NOWRT, | SHR,   | PIC,0                    |                     |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|--------|------------------|--------|--------------------------|---------------------|
|    |    |    |    |    |    | 20 | 3A | 65 | 7A | 69 | 73 | 20 | 2C | 08 | 0099A  | P.AGK:           | .ASCII | <8>\, size: \            |                     |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 08 | 009A3  | P.AGL:           | .ASCII | <8>\!UL bits\            |                     |
|    |    |    |    | 73 |    | 65 | 74 | 79 | 62 | 20 | 4C | 55 | 21 | 09 | 009AC  | P.AGM:           | .ASCII | <9>\!UL bytes\           |                     |
|    |    |    |    | 73 |    | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 20 | 09 | 009B6  | P.AGN:           | .ASCII | <9>\!UL bits\            |                     |
|    |    | 20 |    | 3A |    | 65 | 70 | 79 | 74 | 20 | 6C | 6C | 65 | 0B | 009C0  | P.AGO:           | .ASCII | <11>\cell type: \        |                     |
|    |    | 2C |    | 70 |    | 79 | 74 | 20 | 63 | 69 | 6D | 6F | 74 | 0C | 009CC  | P.AGP:           | .ASCII | <12>\atomic type,\       |                     |
|    |    |    |    |    |    | 20 | 3A | 65 | 7A | 69 | 73 | 20 | 2C | 08 | 009D9  | P.AGQ:           | .ASCII | <8>\, size: \            |                     |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 08 | 009E2  | P.AGR:           | .ASCII | <8>\!UL bits\            |                     |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 09 | 009EB  | P.AGS:           | .ASCII | <9>\!UL bytes\           |                     |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 09 | 009F5  | P.AGT:           | .ASCII | <9>\!UL bits\            |                     |
| 79 | 74 | 20 | 6E | 6F |    | 69 | 74 | 61 | 72 | 65 | 6D | 75 | 6E | 10 | 009FF  | P.AGU:           | .ASCII | <16>\enumeration type\   |                     |
|    |    |    |    |    |    |    |    |    |    |    |    |    |    | 65 | 00A0E  |                  |        |                          |                     |
|    |    |    |    |    |    |    |    | 2C | 43 | 41 | 21 | 28 | 20 | 06 | 00A10  | P.AGV:           | .ASCII | <6>\ (!AC,\              |                     |
|    |    | 2C | 73 | 75 | 6F | 6D | 79 | 6E | 6F | 6E | 61 | 28 | 20 | 0C | 00A17  | P.AGW:           | .ASCII | <12>\ (anonymous,\       |                     |
| 29 | 73 | 74 | 6E | 65 | 6D | 65 | 6C | 65 | 20 | 4C | 55 | 21 | 20 | 0E | 00A24  | P.AGX:           | .ASCII | <14>\!UL elements\       |                     |
|    |    |    |    |    |    | 20 | 3A | 65 | 7A | 69 | 73 | 20 | 2C | 08 | 00A33  | P.AGY:           | .ASCII | <8>\, size: \            |                     |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 08 | 00A3C  | P.AGZ:           | .ASCII | <8>\!UL bits\            |                     |
|    |    |    |    |    |    | 73 | 65 | 74 | 79 | 62 | 20 | 4C | 55 | 21 | 09     | 00A45            | P.AHA: | .ASCII                   | <9>\!UL bytes\      |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 09 | 00A4F  | P.AHB:           | .ASCII | <9>\!UL bits\            |                     |
|    |    | 2C | 65 | 70 | 79 | 74 | 20 | 65 | 72 | 75 | 74 | 63 | 69 | 70 | 0D     | 00A59            | P.AHC: | .ASCII                   | <13>\picture type,\ |
|    |    |    |    |    |    |    |    |    |    | 43 | 41 | 21 | 20 | 04 | 00A67  | P.AHD:           | .ASCII | <4>\!AC\                 |                     |
|    |    |    |    |    | 53 | 41 | 21 | 20 | 3A | 63 | 69 | 70 | 20 | 09 | 00A6C  | P.AHE:           | .ASCII | <9>\pic: !AS\            |                     |
| 20 | 72 | 65 | 74 | 6E | 69 | 6F | 70 | 20 | 64 | 65 | 70 | 79 | 74 | 12 | 00A76  | P.AHF:           | .ASCII | <18>\typed pointer type\ |                     |
|    |    |    |    |    |    |    |    |    |    | 65 | 70 | 79 | 74 |    | 00A85  |                  |        |                          |                     |
|    |    | 20 | 3A | 65 | 70 | 79 | 74 | 20 | 74 | 65 | 67 | 72 | 61 | 74 | 0D     | 00A89            | P.AHG: | .ASCII                   | <13>\target type: \ |
|    |    |    |    | 65 | 70 | 79 | 74 | 20 | 64 | 72 | 6F | 63 | 65 | 72 | 0B     | 00A97            | P.AHH: | .ASCII                   | <11>\record type\   |
|    |    |    |    |    |    |    |    | 2C | 43 | 41 | 21 | 28 | 20 | 06 | 00AA3  | P.AHI:           | .ASCII | <6>\ (!AC,\              |                     |



|    |    |    |    |    |    |    |    |    |    |    |    |    |    |       |        |        |                                    |                            |  |  |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|--------|--------|------------------------------------|----------------------------|--|--|
| 74 | 6E | 2C | 73 | 75 | 6F | 6D | 79 | 6E | 6F | 6E | 61 | 28 | 20 | 0C    | 00AAA  | P.AHJ: | .ASCII                             | <12>\ (anonymous,\         |  |  |
|    |    | 2C | 73 | 75 | 6F | 6D | 79 | 6E | 6F | 6E | 61 | 28 | 20 | 0C    | 00AB7  | P.AHK: | .ASCII                             | <12>\ (anonymous,\         |  |  |
|    |    | 65 | 6E | 6F | 70 | 6D | 6F | 63 | 20 | 4C | 55 | 21 | 20 | 10    | 00AC4  | P.AHL: | .ASCII                             | <16>\ !UL components\      |  |  |
|    |    |    |    |    |    |    |    |    |    |    |    |    |    | 73    | 00AD3  |        |                                    |                            |  |  |
|    |    |    |    |    |    | 20 | 3A | 65 | 7A | 69 | 73 | 20 | 2C | 08    | 00AD5  | P.AHM: | .ASCII                             | <8>\ size: \               |  |  |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 08    | 00ADE  | P.AHN: | .ASCII                             | <8>\ !UL bits\             |  |  |
|    |    |    |    |    | 73 | 65 | 74 | 79 | 62 | 20 | 4C | 55 | 21 | 09    | 00AE7  | P.AHO: | .ASCII                             | <9>\ !UL bytes\            |  |  |
|    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 20 | 09    | 00AF1  | P.AHP: | .ASCII                             | <9>\ !UL bits\             |  |  |
|    |    |    |    |    |    | 65 | 70 | 79 | 74 | 20 | 74 | 65 | 73 | 08    | 00AFB  | P.AHQ: | .ASCII                             | <8>\ set type\             |  |  |
|    |    |    |    |    |    | 20 | 3A | 65 | 7A | 69 | 73 | 20 | 2C | 08    | 00B04  | P.AHR: | .ASCII                             | <8>\ size: \               |  |  |
|    |    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 08    | 00B0D  | P.AHS: | .ASCII                             | <8>\ !UL bits\             |  |  |
|    |    |    |    |    | 73 | 65 | 74 | 79 | 62 | 20 | 4C | 55 | 21 | 09    | 00B16  | P.AHT: | .ASCII                             | <9>\ !UL bytes\            |  |  |
|    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 20 | 09    | 00B20  | P.AHU: | .ASCII                             | <9>\ !UL bits\             |  |  |
| 20 | 3A | 65 | 70 | 74 | 79 | 74 | 20 | 74 | 6E | 65 | 72 | 61 | 70 | 0D    | 00B2A  | P.AHV: | .ASCII                             | <13>\ parent type: \       |  |  |
| 65 | 70 | 79 | 74 | 20 | 65 | 67 | 6E | 61 | 72 | 62 | 75 | 73 | 0D | 00B38 | P.AHW: | .ASCII | <13>\ subrange type\               |                            |  |  |
|    |    |    |    |    | 20 | 3A | 65 | 7A | 69 | 73 | 20 | 2C | 08 | 00B46 | P.AHX: | .ASCII | <8>\ size: \                       |                            |  |  |
|    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 08 | 00B4F | P.AHY: | .ASCII | <8>\ !UL bits\                     |                            |  |  |
|    |    |    |    |    | 73 | 65 | 74 | 79 | 62 | 20 | 4C | 55 | 21 | 09    | 00B58  | P.AHZ: | .ASCII                             | <9>\ !UL bytes\            |  |  |
|    |    |    |    |    | 73 | 74 | 69 | 62 | 20 | 4C | 55 | 21 | 20 | 09    | 00B62  | P.AIA: | .ASCII                             | <9>\ !UL bits\             |  |  |
|    |    |    |    |    |    | 3A | 65 | 67 | 6E | 61 | 72 | 20 | 2C | 08    | 00B6C  | P.AIB: | .ASCII                             | <8>\ range: \              |  |  |
|    |    |    |    |    |    |    |    |    | 2E | 4C | 53 | 21 | 20 | 05    | 00B75  | P.AIC: | .ASCII                             | <5>\ !SL\                  |  |  |
|    |    |    |    |    |    |    |    |    |    | 4C | 53 | 21 | 2E | 04    | 00B7B  | P.AID: | .ASCII                             | <4>\ !SL\                  |  |  |
| 20 | 20 | 3A | 65 | 70 | 79 | 74 | 20 | 74 | 6E | 65 | 72 | 61 | 70 | 0D    | 00B80  | P.AIE: | .ASCII                             | <13>\ parent type: \       |  |  |
| 6C | 61 | 6D | 72 | 6F | 66 | 20 | 6C | 6F | 62 | 6F | 63 | 20 | 1C | 00B8E | P.AIF: | .ASCII | <28>\ cobol formal parameter type\ |                            |  |  |
| 65 | 70 | 79 | 74 | 20 | 72 | 65 | 74 | 65 | 6D | 61 | 72 | 61 | 70 | 00B9D |        |        |                                    |                            |  |  |
| 74 | 69 | 20 | 61 | 74 | 61 | 64 | 20 | 73 | 73 | 69 | 6C | 62 | 14 | 00BAB | P.AIG: | .ASCII | <20>\ bliss data item type\        |                            |  |  |
|    |    |    |    |    |    |    |    |    |    |    |    |    |    | 6D    | 00BBA  |        |                                    |                            |  |  |
| 79 | 74 | 20 | 64 | 6C | 65 | 69 | 66 | 20 | 73 | 73 | 69 | 6C | 62 | 10    | 00BC0  | P.AIH: | .ASCII                             | <16>\ bliss field type\    |  |  |
|    |    |    |    |    |    |    |    |    |    |    |    |    |    | 65    | 70     | 00BCF  |                                    |                            |  |  |
| 65 | 70 | 79 | 74 | 20 | 61 | 74 | 61 | 64 | 20 | 65 | 6C | 69 | 66 | 0F    | 00BD1  | P.AII: | .ASCII                             | <15>\ file data type,\     |  |  |
|    |    |    |    |    |    |    |    |    |    |    |    |    |    | 2C    | 00BE0  |        |                                    |                            |  |  |
|    |    |    |    |    |    |    |    |    |    | 43 | 41 | 21 | 20 | 04    | 00BE1  | P.AIJ: | .ASCII                             | <4>\ !AC\                  |  |  |
| 65 | 20 | 3A | 65 | 70 | 79 | 74 | 20 | 74 | 65 | 67 | 72 | 61 | 74 | 0D    | 00BE6  | P.AIK: | .ASCII                             | <13>\ target type: \       |  |  |
| 74 | 6E | 69 | 6F | 70 | 20 | 64 | 65 | 70 | 79 | 74 | 6E | 75 | 14 | 00BF4 | P.AIL: | .ASCII | <20>\ untyped pointer type\        |                            |  |  |
|    |    |    |    |    |    |    |    |    |    | 65 | 70 | 79 | 74 | 20    | 72     | 00C03  |                                    |                            |  |  |
| 64 | 61 | 20 | 65 | 6C | 69 | 66 | 20 | 64 | 72 | 6F | 63 | 65 | 72 | 13    | 00C09  | P.AIM: | .ASCII                             | <19>\ record file address\ |  |  |
|    |    |    |    |    |    |    |    |    |    | 73 | 73 | 65 | 72 | 64    | 00C18  |        |                                    |                            |  |  |
| 20 | 65 | 76 | 69 | 74 | 61 | 6C | 65 | 72 | 20 | 66 | 6C | 65 | 73 | 13    | 00C1D  | P.AIN: | .ASCII                             | <19>\ self relative label\ |  |  |
|    |    |    |    |    |    |    |    |    |    | 6C | 65 | 62 | 61 | 6C    | 00C2C  |        |                                    |                            |  |  |
| 74 | 79 | 62 | 20 | 2C | 65 | 70 | 79 | 74 | 20 | 61 | 65 | 72 | 61 | 0A    | 00C31  | P.AIO: | .ASCII                             | <10>\ area type,\          |  |  |
|    |    |    |    |    |    |    |    |    |    | 6C | 65 | 62 | 61 | 10    | 00C3C  | P.AIP: | .ASCII                             | <16>\ size: !UL bytes\     |  |  |
|    |    |    |    |    |    |    |    |    |    | 73 | 65 | 73 | 65 | 00C4B |        |        |                                    |                            |  |  |
| 3A | 73 | 2C | 65 | 70 | 79 | 74 | 20 | 74 | 65 | 73 | 66 | 66 | 6F | 0C    | 00C4D  | P.AIQ: | .ASCII                             | <12>\ offset type,\        |  |  |
|    |    | 73 | 65 | 72 | 64 | 64 | 61 | 20 | 65 | 74 | 79 | 62 | 20 | 13    | 00C5A  | P.AIR: | .ASCII                             | <19>\ byte address: !XL,\  |  |  |
|    |    |    |    |    |    |    |    |    |    | 2C | 4C | 58 | 21 | 20    | 00C69  |        |                                    |                            |  |  |
| 74 | 79 | 62 | 20 | 4C | 55 | 21 | 20 | 3A | 65 | 7A | 69 | 73 | 20 | 10    | 00C6E  | P.AIS: | .ASCII                             | <16>\ size: !UL bytes\     |  |  |
|    |    |    |    |    |    |    |    |    |    |    |    |    |    | 73    | 65     | 00C7D  |                                    |                            |  |  |
|    |    | 65 | 70 | 79 | 74 | 20 | 64 | 69 | 6C | 61 | 76 | 6E | 69 | 0C    | 00C7F  | P.AIT: | .ASCII                             | <12>\ invalid type\        |  |  |

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

OFFC 00000 SHOW\_SYMBOL TYPE:

|              |             |       |                                      |  |
|--------------|-------------|-------|--------------------------------------|--|
| 5B 00000000' | EF 9E 00002 | WORD  | Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 |  |
| 5A 00000000G | 00 9E 00009 | MOVAB | DBG\$ PRINT_FLAG, R11                |  |
|              |             | MOVAB | DBG\$PRINT_CONTROL, R10              |  |

: 2670







|           |      |      |    |       |            |                             |                  |      |
|-----------|------|------|----|-------|------------|-----------------------------|------------------|------|
| 20        |      | 55   | D6 | 000C2 | INCL       | R5                          |                  |      |
|           |      | 52   | D1 | 000C4 | CMPL       | R2, #32                     |                  |      |
|           |      | 08   | 14 | 000C7 | BGTR       | 4\$                         |                  |      |
|           | FD24 | 52   | DD | 000C9 | PUSHL      | R2                          |                  |      |
|           |      | C7   | 9F | 000CB | PUSHAB     | P.AGL                       |                  |      |
|           |      | 16   | 11 | 000CF | BRB        | 6\$                         |                  |      |
|           |      | 53   | D5 | 000D1 | 4\$: TSTL  | BYTES                       |                  |      |
|           |      | 09   | 13 | 000D3 | BEQL       | 5\$                         |                  |      |
|           | FD2D | 53   | DD | 000D5 | PUSHL      | BYTES                       |                  |      |
| 68        |      | C7   | 9F | 000D7 | PUSHAB     | P.AGM                       |                  |      |
| 09        |      | 02   | FB | 000DB | CALLS      | #2, DBG\$PRINT              |                  |      |
|           |      | 55   | E9 | 000DE | 5\$: BLBC  | R5, 7\$                     |                  |      |
|           | FD37 | 54   | DD | 000E1 | PUSHL      | BITS                        |                  |      |
| 68        |      | C7   | 9F | 000E3 | PUSHAB     | P.AGN                       |                  |      |
| 69        |      | 02   | FB | 000E7 | 6\$: CALLS | #2, DBG\$PRINT              |                  |      |
|           |      | 00   | FB | 000EA | 7\$: CALLS | #0, DBG\$NEWLINE            |                  | 2745 |
|           |      | 04   | DD | 000ED | PUSHL      | #4                          |                  | 2746 |
|           |      | 02   | DD | 000EF | PUSHL      | #2                          |                  |      |
| 6A        |      | 02   | FB | 000F1 | CALLS      | #2, DBG\$PRINT_CONTROL      |                  |      |
|           |      | 7E   | D4 | 000F4 | CLRL       | -(SP)                       |                  | 2747 |
|           | FD41 | C7   | 9F | 000F6 | PUSHAB     | P.AGO                       |                  |      |
| 68        |      | 02   | FB | 000FA | CALLS      | #2, DBG\$PRINT              |                  |      |
| 6B        |      | 01   | D0 | 000FD | MOVL       | #1, DBG\$_PRINT_FLAG        |                  | 2748 |
|           | 10   | AE   | DD | 00100 | PUSHL      | CELLTYPE                    |                  | 2749 |
|           |      | 0444 | 31 | 00103 | BRW        | 52\$                        |                  |      |
|           |      | 7E   | D4 | 00106 | 8\$: CLRL  | -(SP)                       |                  | 2755 |
|           | FD4D | C7   | 9F | 00108 | PUSHAB     | P.AGP                       |                  |      |
| 68        |      | 02   | FB | 0010C | CALLS      | #2, DBG\$PRINT              |                  |      |
|           | 34   | AE   | 9F | 0010F | PUSHAB     | BITSIZE                     |                  | 2756 |
|           | 18   | AE   | 9F | 00112 | PUSHAB     | TYPECODE                    |                  |      |
|           | 10   | AE   | DD | 00115 | PUSHL      | TYPEID                      |                  |      |
| 00000000G | 00   | 03   | FB | 00118 | CALLS      | #3, DBG\$STA_TYP_ATOMIC     |                  |      |
|           |      | 14   | AE | DD    | 0011F      | TYPECODE                    |                  | 2757 |
| F519      | CF   | 01   | FB | 00122 | CALLS      | #1, DBG\$TRANSLATE_TYPECODE |                  |      |
|           | 52   | 34   | AE | D0    | 00127      | MOVL                        | BITSIZE, R2      | 2758 |
|           |      | 55   | 13 | 0012B | BEQL       | 13\$                        |                  |      |
| 54        | 53   | 52   | 08 | C7    | 0012D      | DIVL3                       | #8, R2, BYTES    |      |
|           | 52   | 03   | 00 | EF    | 00131      | EXTZV                       | #0, #3, R2, BITS |      |
|           |      |      | 7E | D4    | 00136      | CLRL                        | -(SP)            |      |
|           |      |      | C7 | 9F    | 00138      | PUSHAB                      | P.AGO            |      |
| 68        |      |      | 02 | FB    | 0013C      | CALLS                       | #2, DBG\$PRINT   |      |
|           |      |      | 55 | D4    | 0013F      | CLRL                        | R5               |      |
|           |      |      | 54 | D5    | 00141      | TSTL                        | BITS             |      |
|           |      |      | 0F | 13    | 00143      | BEQL                        | 9\$              |      |
|           |      |      | 55 | D6    | 00145      | INCL                        | R5               |      |
| 20        |      |      | 52 | D1    | 00147      | CMPL                        | R2, #32          |      |
|           |      |      | 08 | 14    | 0014A      | BGTR                        | 9\$              |      |
|           |      |      | 52 | DD    | 0014C      | PUSHL                       | R2               |      |
|           | FD63 |      | C7 | 9F    | 0014E      | PUSHAB                      | P.AGR            |      |
|           |      |      | 16 | 11    | 00152      | BRB                         | 11\$             |      |
|           |      |      | 53 | D5    | 00154      | 9\$: TSTL                   | BYTES            |      |
|           |      |      | 09 | 13    | 00156      | BEQL                        | 10\$             |      |
|           |      |      | 53 | DD    | 00158      | PUSHL                       | BYTES            |      |
|           | FD6C |      | C7 | 9F    | 0015A      | PUSHAB                      | P.AGS            |      |
| 68        |      |      | 02 | FB    | 0015E      | CALLS                       | #2, DBG\$PRINT   |      |
| 1E        |      |      | 55 | E9    | 00161      | 10\$: BLBC                  | R5, 13\$         |      |
|           |      |      | 54 | DD    | 00164      | PUSHL                       | BITS             |      |



|           |    |      |      |    |       |        |                          |      |
|-----------|----|------|------|----|-------|--------|--------------------------|------|
|           |    | FD76 | C7   | 9F | 00166 | PUSHAB | P.AGT                    |      |
|           |    |      | 0446 | 31 | 0016A | BRW    | 61\$                     |      |
|           |    | 18   | AE   | 9F | 0016D | PUSHAB | DSCADDR                  | 2763 |
|           |    | 0C   | AE   | DD | 00170 | PUSHL  | TYPEID                   |      |
| 00000000G | 00 |      | 02   | FB | 00173 | CALLS  | #2, DBG\$STA_TYP_DESCR   |      |
|           |    | 18   | AE   | DD | 0017A | PUSHL  | DSCADDR                  | 2764 |
| F1C3      | CF |      | 01   | FB | 0017D | CALLS  | #1, DBG\$TRANSLATE_DESCR |      |
|           |    |      | 0431 | 31 | 00182 | BRW    | 62\$                     | 2737 |
|           |    |      | 7E   | D4 | 00185 | CLRL   | -(SP)                    | 2769 |
|           |    | FD80 | C7   | 9F | 00187 | PUSHAB | P.AGU                    |      |
|           | 68 |      | 02   | FB | 0018B | CALLS  | #2, DBG\$PRINT           |      |
|           |    | 34   | AE   | 9F | 0018E | PUSHAB | BITSIZE                  | 2770 |
|           |    | 2C   | AE   | 9F | 00191 | PUSHAB | ELTVECPTR                |      |
|           |    | 34   | AE   | 9F | 00194 | PUSHAB | NELTS                    |      |
|           | 52 | 14   | AE   | D0 | 00197 | MOVL   | TYPEID, R2               |      |
|           |    |      | 52   | DD | 0019B | PUSHL  | R2                       |      |
| 00000000G | 00 |      | 04   | FB | 0019D | CALLS  | #4, DBG\$STA_TYP_ENUM    |      |
|           |    | 0C   | A2   | DD | 001A4 | PUSHL  | 12(R2)                   | 2771 |
| 00000000G | 00 |      | 01   | FB | 001A7 | CALLS  | #1, DBG\$GET_DST_NAME    |      |
|           | 52 |      | 50   | D0 | 001AE | MOVL   | R0, NAME                 |      |
|           |    |      | 62   | 95 | 001B1 | TSTB   | (NAME)                   | 2772 |
|           |    |      | 08   | 13 | 001B3 | BEQL   | 15\$                     |      |
|           |    |      | 52   | DD | 001B5 | PUSHL  | NAME                     | 2774 |
|           |    | FD91 | C7   | 9F | 001B7 | PUSHAB | P.AGV                    |      |
|           |    |      | 06   | 11 | 001BB | BRB    | 16\$                     |      |
|           |    |      | 7E   | D4 | 001BD | CLRL   | -(SP)                    | 2776 |
|           |    | FD98 | C7   | 9F | 001BF | PUSHAB | P.AGW                    |      |
|           | 68 |      | 02   | FB | 001C3 | CALLS  | #2, DBG\$PRINT           |      |
|           |    | 2C   | AE   | DD | 001C6 | PUSHL  | NELTS                    | 2778 |
|           |    | FDA5 | C7   | 9F | 001C9 | PUSHAB | P.AGX                    |      |
|           | 68 |      | 02   | FB | 001CD | CALLS  | #2, DBG\$PRINT           |      |
|           | 52 | 34   | AE   | D0 | 001D0 | MOVL   | BITSIZE, R2              | 2779 |
|           |    |      | AC   | 13 | 001D4 | BEQL   | 13\$                     |      |
|           | 52 |      | 08   | C7 | 001D6 | DIVL3  | #8, R2, BYTES            |      |
| 54        | 53 |      | 00   | EF | 001DA | EXTZV  | #0, #3, R2, BITS         |      |
|           | 52 |      | 7E   | D4 | 001DF | CLRL   | -(SP)                    |      |
|           | 03 |      | C7   | 9F | 001E1 | PUSHAB | P.AGY                    |      |
|           |    | 68   | 02   | FB | 001E5 | CALLS  | #2, DBG\$PRINT           |      |
|           |    |      | 55   | D4 | 001E8 | CLRL   | R5                       |      |
|           |    |      | 54   | D5 | 001EA | TSTL   | BITS                     |      |
|           |    |      | 0F   | 13 | 001EC | BEQL   | 17\$                     |      |
|           |    |      | 55   | D6 | 001EE | INCL   | R5                       |      |
|           | 20 |      | 52   | D1 | 001F0 | CMPL   | R2, #32                  |      |
|           |    |      | 08   | 14 | 001F3 | BGTR   | 17\$                     |      |
|           |    |      | 52   | DD | 001F5 | PUSHL  | R2                       |      |
|           |    | FDBD | C7   | 9F | 001F7 | PUSHAB | P.AGZ                    |      |
|           |    |      | 19   | 11 | 001FB | BRB    | 20\$                     |      |
|           |    |      | 53   | D5 | 001FD | TSTL   | BYTES                    |      |
|           |    |      | 09   | 13 | 001FF | BEQL   | 18\$                     |      |
|           |    |      | 53   | DD | 00201 | PUSHL  | BYTES                    |      |
|           |    | FDC6 | C7   | 9F | 00203 | PUSHAB | P.AHA                    |      |
|           | 68 |      | 02   | FB | 00207 | CALLS  | #2, DBG\$PRINT           |      |
|           | 03 |      | 55   | E8 | 0020A | BLBS   | R5, 19\$                 |      |
|           |    |      | 03A6 | 31 | 0020D | BRW    | 62\$                     |      |
|           |    |      | 54   | DD | 00210 | PUSHL  | BITS                     |      |
|           |    | FDD0 | C7   | 9F | 00212 | PUSHAB | P.AHB                    |      |
|           |    |      | 039A | 31 | 00216 | BRW    | 61\$                     |      |



|           |    |          |          |       |       |       |        |                            |      |
|-----------|----|----------|----------|-------|-------|-------|--------|----------------------------|------|
|           |    |          | 7E       | D4    | 00219 | 21\$: | CLRL   | -(SP)                      | 2784 |
|           |    | FDDA     | C7       | 9F    | 0021B |       | PUSHAB | P.AHC                      |      |
|           | 68 |          | 02       | FB    | 0021F |       | CALLS  | #2, DBG\$PRINT             |      |
|           |    | 1C       | AE       | 9F    | 00222 |       | PUSHAB | PSCALE                     | 2785 |
|           |    | 24       | AE       | 9F    | 00225 |       | PUSHAB | PICTVAL                    |      |
|           |    | 2C       | AE       | 9F    | 00228 |       | PUSHAB | PICTPTR                    |      |
|           |    | 54       | AE       | 9F    | 0022B |       | PUSHAB | LANGCODE                   |      |
|           |    | 18       | AE       | DD    | 0022E |       | PUSHL  | TYPEID                     |      |
| 00000000G | 00 |          | 05       | FB    | 00231 |       | CALLS  | #5, DBG\$STA_TYP_PICT      |      |
|           |    | 48       | AE       | DD    | 00238 |       | PUSHL  | LANGCODE                   | 2786 |
| 00000000G | 00 |          | 01       | FB    | 0023B |       | CALLS  | #1, DBG\$LANGUAGE          |      |
|           | 54 |          | 50       | D0    | 00242 |       | MOVL   | R0, LANGUAGE               |      |
|           |    |          | 54       | DD    | 00245 |       | PUSHL  | LANGUAGE                   | 2787 |
|           |    | FDE8     | C7       | 9F    | 00247 |       | PUSHAB | P.AHD                      |      |
|           | 68 |          | 02       | FB    | 0024B |       | CALLS  | #2, DBG\$PRINT             |      |
|           | 53 | 24       | AE       | D0    | 0024E |       | MOVL   | PICTPTR, R3                | 2788 |
|           |    |          | 63       | 95    | 00252 |       | TSTB   | (R3)                       |      |
|           |    |          | 77       | 13    | 00254 |       | BEQL   | 24\$                       |      |
| 5C        | AE | 020E0000 | 8F       | D0    | 00256 |       | MOVL   | #34471936, PICT_DESC       | 2791 |
|           |    | 60       | AE       | D4    | 0025E |       | CLRL   | PICT_DESC+4                |      |
|           | 50 | FF       | A3       | 9E    | 00261 |       | MOVAB  | -1(R3), PICT_LENPTR        | 2792 |
|           | 50 |          | 60       | 9A    | 00265 |       | MOVZBL | (PICT_LENPTR), PICT_LENGTH | 2793 |
| 54        | 50 |          | 02       | C7    | 00268 |       | DIVL3  | #2, PICT_LENGTH, R4        | 2794 |
|           |    |          | 52       | D4    | 0026C |       | CLRL   | PICT_INDEX                 |      |
|           |    |          | 45       | 11    | 0026E |       | BRB    | 23\$                       |      |
|           | 54 | AE       | 020E0000 | 8F    | D0    | 22\$: | MOVL   | #34471936, DUPL_DESC       | 2801 |
|           |    | 58       | AE       | D4    | 00278 |       | CLRL   | DUPL_DESC+4                |      |
|           |    | FE       | A342     | 3F    | 0027B |       | PUSHAB | -2(R3)[PICT_INDEX]         | 2804 |
|           | 04 | AE       | 9E       | 9A    | 0027F |       | MOVZBL | @(SP)+, 4(SP)              |      |
|           |    | 04       | AE       | 9F    | 00283 |       | PUSHAB | 4(SP)                      |      |
|           |    | FF       | A342     | 3F    | 00286 |       | PUSHAB | -1(R3)[PICT_INDEX]         | 2803 |
|           | 04 | AE       | 9E       | 9A    | 0028A |       | MOVZBL | @(SP)+, 4(SP)              |      |
|           |    | 04       | AE       | 9F    | 0028E |       | PUSHAB | 4(SP)                      |      |
|           |    | 5C       | AE       | 9F    | 00291 |       | PUSHAB | DUPL_DESC                  | 2802 |
| 00000000G | 00 |          | 03       | FB    | 00294 |       | CALLS  | #3, STR\$DUPL_CHAR         |      |
|           |    | 54       | AE       | 9F    | 0029B |       | PUSHAB | DUPL_DESC                  | 2805 |
|           |    | 60       | AE       | 9F    | 0029E |       | PUSHAB | PICT_DESC                  |      |
|           |    | 64       | AE       | 9F    | 002A1 |       | PUSHAB | PICT_DESC                  |      |
| 00000000G | 00 |          | 03       | FB    | 002A4 |       | CALLS  | #3, STR\$CONCAT            |      |
|           |    | 54       | AE       | 9F    | 002AB |       | PUSHAB | DUPL_DESC                  | 2806 |
| 00000000G | 00 |          | 01       | FB    | 002AE |       | CALLS  | #1, STR\$FREE1_DX          |      |
| B7        | 52 |          | 54       | F3    | 002B5 | 23\$: | AOBLEQ | R4, PICT_INDEX, 22\$       | 2794 |
|           |    | 5C       | AE       | 9F    | 002B9 |       | PUSHAB | PICT_DESC                  | 2809 |
|           |    | FDED     | C7       | 9F    | 002BC |       | PUSHAB | P.AHE                      |      |
|           | 68 |          | 02       | FB    | 002C0 |       | CALLS  | #2, DBG\$PRINT             |      |
|           |    | 5C       | AE       | 9F    | 002C3 |       | PUSHAB | PICT_DESC                  | 2810 |
| 00000000G | 00 |          | 01       | FB    | 002C6 |       | CALLS  | #1, STR\$FREE1_DX          |      |
|           |    | 02E6     | 31       | 002CD | 24\$: | BRW   | 62\$   |                            | 2737 |
|           |    | 7E       | D4       | 002D0 | 25\$: | CLRL  | -(SP)  |                            | 2817 |
|           |    | FDF7     | C7       | 9F    | 002D2 |       | PUSHAB | P.AHF                      |      |
|           | 68 |          | 02       | FB    | 002D6 |       | CALLS  | #2, DBG\$PRINT             |      |
|           | 69 |          | 00       | FB    | 002D9 |       | CALLS  | #0, DBG\$NEWLINE           | 2818 |
|           |    | 44       | AE       | 9F    | 002DC |       | PUSHAB | REFTYPID                   | 2819 |
|           |    | 0C       | AE       | DD    | 002DF |       | PUSHL  | TYPEID                     |      |
| 00000000G | 00 |          | 02       | FB    | 002E2 |       | CALLS  | #2, DBG\$STA_TYP_TYPEDPTR  |      |
|           |    |          | 04       | DD    | 002E9 |       | PUSHL  | #4                         | 2820 |
|           |    |          | 02       | DD    | 002EB |       | PUSHL  | #2                         |      |



|           |      |      |    |          |        |                         |       |
|-----------|------|------|----|----------|--------|-------------------------|-------|
|           | 6A   | 02   | FB | 002ED    | CALLS  | #2, DBG\$PRINT_CONTROL  |       |
|           |      | 7E   | D4 | 002F0    | CLRL   | -(SP)                   | 2821  |
|           | FE0A | C7   | 9F | 002F2    | PUSHAB | P.AHG                   |       |
|           |      | 0248 | 31 | 002F6    | BRW    | 51\$                    |       |
|           |      | 7E   | D4 | 002F9    | CLRL   | -(SP)                   | 2829  |
|           | FE18 | C7   | 9F | 002FB    | PUSHAB | P.AHH                   |       |
|           | 68   | 02   | FB | 002FF    | CALLS  | #2, DBG\$PRINT          |       |
|           | 53   | 08   | AE | D0 00302 | MOVL   | TYPEID, R3              | 2836  |
|           | 52   | 0C   | A3 | D0 00306 | MOVL   | 12(R3), DSTPTR          |       |
| 80        | 8F   | 02   | A2 | 91 0030A | CMPB   | 2(DSTPTR), #128         | 2837  |
|           |      | 21   | 12 | 0030F    | BNEQ   | 28\$                    |       |
|           |      | 0C   | A3 | DD 00311 | PUSHL  | 12(R3)                  | 2840  |
| 00000000G | 00   | 01   | FB | 00314    | CALLS  | #1, DBG\$GET_DST_NAME   |       |
|           | 52   | 50   | D0 | 0031B    | MOVL   | R0, NAME                |       |
|           |      | 62   | 95 | 0031E    | TSTB   | (NAME)                  | 2841  |
|           |      | 08   | 13 | 00320    | BEQL   | 27\$                    |       |
|           |      | 52   | DD | 00322    | PUSHL  | NAME                    | 2843  |
|           | FE24 | C7   | 9F | 00324    | PUSHAB | P.AHI                   |       |
|           |      | 0E   | 11 | 00328    | BRB    | 29\$                    |       |
|           |      | 7E   | D4 | 0032A    | CLRL   | -(SP)                   | 2845  |
|           | FE2B | C7   | 9F | 0032C    | PUSHAB | P.AHJ                   |       |
|           |      | 06   | 11 | 00330    | BRB    | 29\$                    |       |
|           |      | 7E   | D4 | 00332    | CLRL   | -(SP)                   | 2850  |
|           | FE38 | C7   | 9F | 00334    | PUSHAB | P.AHK                   |       |
|           | 68   | 02   | FB | 00338    | CALLS  | #2, DBG\$PRINT          |       |
|           |      | 34   | AE | 9F 0033B | PUSHAB | BITSIZE                 | 2852  |
|           |      | 2C   | AE | 9F 0033E | PUSHAB | ELTVECPTR               |       |
|           |      | 34   | AE | 9F 00341 | PUSHAB | NELTS                   |       |
|           |      | 53   | DD | 00344    | PUSHL  | R3                      |       |
| 00000000G | 00   | 04   | FB | 00346    | CALLS  | #4, DBG\$STA_TYP_RECORD |       |
|           |      | 2C   | AE | DD 0034D | PUSHL  | NELTS                   | 2853  |
|           | FE45 | C7   | 9F | 00350    | PUSHAB | P.AHL                   |       |
|           | 68   | 02   | FB | 00354    | CALLS  | #2, DBG\$PRINT          |       |
|           | 52   | 34   | AE | D0 00357 | MOVL   | BITSIZE, R2             | 2854  |
|           |      | 40   | 13 | 0035B    | BEQL   | 33\$                    |       |
|           |      | 08   | C7 | 0035D    | DIVL3  | #8, R2, BYTES           |       |
| 54        | 53   | 00   | EF | 00361    | EXTZV  | #0, #3, R2, BITS        |       |
|           | 52   | 7E   | D4 | 00366    | CLRL   | -(SP)                   |       |
|           | 03   | C7   | 9F | 00368    | PUSHAB | P.AHM                   |       |
|           |      | 68   | 02 | FB 0036C | CALLS  | #2, DBG\$PRINT          |       |
|           |      |      | 55 | D4 0036F | CLRL   | R5                      |       |
|           |      |      | 54 | D5 00371 | TSTL   | BITS                    |       |
|           |      |      | 0F | 13 00373 | BEQL   | 30\$                    |       |
|           |      |      | 55 | D6 00375 | INCL   | R5                      |       |
|           | 20   | 52   | D1 | 00377    | CMPL   | R2, #32                 |       |
|           |      | 08   | 14 | 0037A    | BGTR   | 30\$                    |       |
|           |      | 52   | DD | 0037C    | PUSHL  | R2                      |       |
|           |      | FE5F | C7 | 9F 0037E | PUSHAB | P.AHN                   |       |
|           |      | 16   | 11 | 00382    | BRB    | 32\$                    |       |
|           |      | 53   | D5 | 00384    | TSTL   | BYTES                   | 30\$: |
|           |      | 09   | 13 | 00386    | BEQL   | 31\$                    |       |
|           |      | 53   | DD | 00388    | PUSHL  | BYTES                   |       |
|           |      | FE68 | C7 | 9F 0038A | PUSHAB | P.AHO                   |       |
|           | 68   | 02   | FB | 0038E    | CALLS  | #2, DBG\$PRINT          |       |
|           | 09   | 55   | E9 | 00391    | BLBC   | R5, 33\$                | 31\$: |
|           |      | 54   | DD | 00394    | PUSHL  | BITS                    |       |
|           |      | FE72 | C7 | 9F 00396 | PUSHAB | P.AHP                   |       |



| PC | Op | Op2 | Op3 | Op4 | Op5 | Op6 | Op7 | Op8 | Op9 | Op10 | Op11 | Op12 | Op13 | Op14 | Op15 | Op16 | Op17 | Op18 | Op19 | Op20 | Op21 | Op22 | Op23 | Op24 | Op25 | Op26 | Op27 | Op28 | Op29 | Op30 | Op31 | Op32 | Op33 | Op34 | Op35 | Op36 | Op37 | Op38 | Op39 | Op40 | Op41 | Op42 | Op43 | Op44 | Op45 | Op46 | Op47 | Op48 | Op49 | Op50 | Op51 | Op52 | Op53 | Op54 | Op55 | Op56 | Op57 | Op58 | Op59 | Op60 | Op61 | Op62 | Op63 | Op64 | Op65 | Op66 | Op67 | Op68 | Op69 | Op70 | Op71 | Op72 | Op73 | Op74 | Op75 | Op76 | Op77 | Op78 | Op79 | Op80 | Op81 | Op82 | Op83 | Op84 | Op85 | Op86 | Op87 | Op88 | Op89 | Op90 | Op91 | Op92 | Op93 | Op94 | Op95 | Op96 | Op97 | Op98 | Op99 | Op100 | Op101 | Op102 | Op103 | Op104 | Op105 | Op106 | Op107 | Op108 | Op109 | Op110 | Op111 | Op112 | Op113 | Op114 | Op115 | Op116 | Op117 | Op118 | Op119 | Op120 | Op121 | Op122 | Op123 | Op124 | Op125 | Op126 | Op127 | Op128 | Op129 | Op130 | Op131 | Op132 | Op133 | Op134 | Op135 | Op136 | Op137 | Op138 | Op139 | Op140 | Op141 | Op142 | Op143 | Op144 | Op145 | Op146 | Op147 | Op148 | Op149 | Op150 | Op151 | Op152 | Op153 | Op154 | Op155 | Op156 | Op157 | Op158 | Op159 | Op160 | Op161 | Op162 | Op163 | Op164 | Op165 | Op166 | Op167 | Op168 | Op169 | Op170 | Op171 | Op172 | Op173 | Op174 | Op175 | Op176 | Op177 | Op178 | Op179 | Op180 | Op181 | Op182 | Op183 | Op184 | Op185 | Op186 | Op187 | Op188 | Op189 | Op190 | Op191 | Op192 | Op193 | Op194 | Op195 | Op196 | Op197 | Op198 | Op199 | Op200 | Op201 | Op202 | Op203 | Op204 | Op205 | Op206 | Op207 | Op208 | Op209 | Op210 | Op211 | Op212 | Op213 | Op214 | Op215 | Op216 | Op217 | Op218 | Op219 | Op220 | Op221 | Op222 | Op223 | Op224 | Op225 | Op226 | Op227 | Op228 | Op229 | Op230 | Op231 | Op232 | Op233 | Op234 | Op235 | Op236 | Op237 | Op238 | Op239 | Op240 | Op241 | Op242 | Op243 | Op244 | Op245 | Op246 | Op247 | Op248 | Op249 | Op250 | Op251 | Op252 | Op253 | Op254 | Op255 | Op256 | Op257 | Op258 | Op259 | Op260 | Op261 | Op262 | Op263 | Op264 | Op265 | Op266 | Op267 | Op268 | Op269 | Op270 | Op271 | Op272 | Op273 | Op274 | Op275 | Op276 | Op277 | Op278 | Op279 | Op280 | Op281 | Op282 | Op283 | Op284 | Op285 | Op286 | Op287 | Op288 | Op289 | Op290 | Op291 | Op292 | Op293 | Op294 | Op295 | Op296 | Op297 | Op298 | Op299 | Op300 | Op301 | Op302 | Op303 | Op304 | Op305 | Op306 | Op307 | Op308 | Op309 | Op310 | Op311 | Op312 | Op313 | Op314 | Op315 | Op316 | Op317 | Op318 | Op319 | Op320 | Op321 | Op322 | Op323 | Op324 | Op325 | Op326 | Op327 | Op328 | Op329 | Op330 | Op331 | Op332 | Op333 | Op334 | Op335 | Op336 | Op337 | Op338 | Op339 | Op340 | Op341 | Op342 | Op343 | Op344 | Op345 | Op346 | Op347 | Op348 | Op349 | Op350 | Op351 | Op352 | Op353 | Op354 | Op355 | Op356 | Op357 | Op358 | Op359 | Op360 | Op361 | Op362 | Op363 | Op364 | Op365 | Op366 | Op367 | Op368 | Op369 | Op370 | Op371 | Op372 | Op373 | Op374 | Op375 | Op376 | Op377 | Op378 | Op379 | Op380 | Op381 | Op382 | Op383 | Op384 | Op385 | Op386 | Op387 | Op388 | Op389 | Op390 | Op391 | Op392 | Op393 | Op394 | Op395 | Op396 | Op397 | Op398 | Op399 | Op400 | Op401 | Op402 | Op403 | Op404 | Op405 | Op406 | Op407 | Op408 | Op409 | Op410 | Op411 | Op412 | Op413 | Op414 | Op415 | Op416 | Op417 | Op418 | Op419 |
|----|----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
|----|----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|



|    |    |    |    |      |      |    |       |        |                             |      |      |
|----|----|----|----|------|------|----|-------|--------|-----------------------------|------|------|
| 54 | 52 | 53 | 00 | 4C   | AE   | 9F | 0044B | PUSHAB | PARENT_TYPE                 | 2880 |      |
|    |    |    | 53 | 18   | AE   | DD | 0044E | PUSHL  | TYPEID                      |      |      |
|    |    |    |    |      | 05   | FB | 00451 | CALLS  | #5, DBG\$STA_TYP_SUBRNG     |      |      |
|    |    |    |    | 34   | AE   | D0 | 00458 | MOVL   | BITSIZE, R3                 |      |      |
|    |    |    |    |      | 40   | 13 | 0045C | BEQL   | 45\$                        |      |      |
|    |    |    | 53 |      | 08   | C7 | 0045E | DIVL3  | #8, R3, BYTES               |      |      |
|    |    |    | 03 |      | 00   | EF | 00462 | EXTZV  | #0, #3, R3, BITS            |      |      |
|    |    |    |    |      | 7E   | D4 | 00467 | CLRL   | -(SP)                       |      |      |
|    |    |    |    | FEC7 | C7   | 9F | 00469 | PUSHAB | P.AHX                       |      |      |
|    |    |    | 68 |      | 02   | FB | 0046D | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    |    |      | 55   | D4 | 00470 | CLRL   | R5                          |      |      |
|    |    |    |    |      | 54   | D5 | 00472 | TSTL   | BITS                        |      |      |
|    |    |    |    |      | 0F   | 13 | 00474 | BEQL   | 42\$                        |      |      |
|    |    |    | 20 |      | 55   | D6 | 00476 | INCL   | R5                          |      |      |
|    |    |    |    |      | 53   | D1 | 00478 | CMPL   | R3, #32                     |      |      |
|    |    |    |    |      | 08   | 14 | 0047B | BGTR   | 42\$                        |      |      |
|    |    |    |    |      | 53   | DD | 0047D | PUSHL  | R3                          |      |      |
|    |    |    |    | FED0 | C7   | 9F | 0047F | PUSHAB | P.AHY                       |      |      |
|    |    |    |    |      | 16   | 11 | 00483 | BRB    | 44\$                        |      |      |
|    |    |    |    |      | 52   | D5 | 00485 | TSTL   | BYTES                       |      |      |
|    |    |    |    |      | 09   | 13 | 00487 | BEQL   | 43\$                        |      |      |
|    |    |    |    |      | 52   | DD | 00489 | PUSHL  | BYTES                       |      |      |
|    |    |    |    | FED9 | C7   | 9F | 0048B | PUSHAB | P.AHZ                       |      |      |
|    |    |    | 68 |      | 02   | FB | 0048F | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    | 09 |      | 55   | E9 | 00492 | BLBC   | R5, 45\$                    |      |      |
|    |    |    |    |      | 54   | DD | 00495 | PUSHL  | BITS                        |      |      |
|    |    |    |    | FEE3 | C7   | 9F | 00497 | PUSHAB | P.AIA                       |      |      |
|    |    |    | 68 |      | 02   | FB | 0049B | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    |    |      | 7E   | D4 | 0049E | CLRL   | -(SP)                       | 2881 |      |
|    |    |    |    | FEED | C7   | 9F | 004A0 | PUSHAB | P.AIB                       |      |      |
|    |    |    | 68 |      | 02   | FB | 004A4 | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    |    | 3C   | BE   | DD | 004A7 | PUSHL  | @LOWPTR                     | 2882 |      |
|    |    |    |    | FEF6 | C7   | 9F | 004AA | PUSHAB | P.AIC                       |      |      |
|    |    |    | 68 |      | 02   | FB | 004AE | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    |    | 38   | BE   | DD | 004B1 | PUSHL  | @HIGHPTR                    | 2883 |      |
|    |    |    |    | FEFC | C7   | 9F | 004B4 | PUSHAB | P.AID                       |      |      |
|    |    |    | 68 |      | 02   | FB | 004B8 | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    | 69 |      | 00   | FB | 004BB | CALLS  | #0, DBG\$NEWLINE            | 2884 |      |
|    |    |    |    |      | 04   | DD | 004BE | PUSHL  | #4                          | 2885 |      |
|    |    |    |    |      | 02   | DD | 004C0 | PUSHL  | #2                          |      |      |
|    |    |    | 6A |      | 02   | FB | 004C2 | CALLS  | #2, DBG\$PRINT_CONTROL      |      |      |
|    |    |    |    |      | 7E   | D4 | 004C5 | CLRL   | -(SP)                       | 2886 |      |
|    |    |    |    | FF01 | C7   | 9F | 004C7 | PUSHAB | P.AIE                       |      |      |
|    |    |    | 68 |      | 02   | FB | 004CB | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    | 6B |      | 01   | D0 | 004CE | MOVL   | #1, DBG\$PRINT_FLAG         | 2887 |      |
|    |    |    |    | 40   | AE   | DD | 004D1 | PUSHL  | PARENT_TYPE                 | 2888 |      |
|    |    |    |    |      | 74   | 11 | 004D4 | BRB    | 52\$                        |      |      |
|    |    |    |    |      | 7E   | D4 | 004D6 | CLRL   | -(SP)                       | 2894 |      |
|    |    |    |    | FF0F | C7   | 9F | 004D8 | PUSHAB | P.AIF                       |      |      |
|    |    |    |    |      | 0080 | 31 | 004DC | BRW    | 55\$                        |      |      |
|    |    |    |    |      | 7E   | D4 | 004DF | CLRL   | -(SP)                       | 2899 |      |
|    |    |    |    | FF2C | C7   | 9F | 004E1 | PUSHAB | P.AIG                       |      |      |
|    |    |    | 68 |      | 02   | FB | 004E5 | CALLS  | #2, DBG\$PRINT              |      |      |
|    |    |    | 52 |      | A2   | D0 | 004E8 | MOVL   | 12(R2), DSTPTR              | 2900 |      |
|    |    |    |    | OC   | 52   | DD | 004EC | PUSHL  | DSTPTR                      | 2901 |      |
|    |    |    |    |      | 01   | FB | 004EE | CALLS  | #1, GET_BLISS_SPECIAL_CASES |      |      |
|    |    |    |    | F451 | CF   | 62 | 11    | 004F3  | BRB                         | 53\$ | 2737 |



|           |      |    |    |       |       |        |                         |      |
|-----------|------|----|----|-------|-------|--------|-------------------------|------|
|           |      | 7E | D4 | 004F5 | 49\$: | CLRL   | -(SP)                   | 2906 |
|           | FF41 | C7 | 9F | 004F7 |       | PUSHAB | P.AIH                   |      |
|           |      | 70 | 11 | 004FB |       | BRB    | 58\$                    |      |
|           |      | 7E | D4 | 004FD | 50\$: | CLRL   | -(SP)                   | 2911 |
|           | FF52 | C7 | 9F | 004FF |       | PUSHAB | P.AII                   |      |
| 68        |      | 02 | FB | 00503 |       | CALLS  | #2, DBG\$PRINT          |      |
|           | 44   | AE | 9F | 00506 |       | PUSHAB | REFTYPID                | 2912 |
|           | 4C   | AE | 9F | 00509 |       | PUSHAB | LANGCODE                |      |
|           | 10   | AE | DD | 0050C |       | PUSHL  | TYPEID                  |      |
| 00000000G | 00   | 03 | FB | 0050F |       | CALLS  | #3, DBG\$STA_TYP_FILE   |      |
|           | 48   | AE | DD | 00516 |       | PUSHL  | LANGCODE                | 2913 |
| 00000000G | 00   | 01 | FB | 00519 |       | CALLS  | #1, DBG\$LANGUAGE       |      |
| 54        |      | 50 | DD | 00520 |       | MOVL   | RO, LANGUAGE            |      |
|           |      | 54 | DD | 00523 |       | PUSHL  | LANGUAGE                | 2914 |
|           | FF62 | C7 | 9F | 00525 |       | PUSHAB | P.AIJ                   |      |
| 68        |      | 02 | FB | 00529 |       | CALLS  | #2, DBG\$PRINT          |      |
|           | 44   | AE | D5 | 0052C |       | TSTL   | REFTYPID                | 2915 |
|           |      | 26 | 13 | 0052F |       | BEQL   | 53\$                    |      |
| 69        |      | 00 | FB | 00531 |       | CALLS  | #0, DBG\$NEWLINE        | 2918 |
|           |      | 04 | DD | 00534 |       | PUSHL  | #4                      | 2919 |
|           |      | 02 | DD | 00536 |       | PUSHL  | #2                      |      |
| 6A        |      | 02 | FB | 00538 |       | CALLS  | #2, DBG\$PRINT_CONTROL  |      |
|           |      | 7E | D4 | 0053B |       | CLRL   | -(SP)                   | 2920 |
|           | FF67 | C7 | 9F | 0053D |       | PUSHAB | P.AIK                   |      |
| 68        |      | 02 | FB | 00541 | 51\$: | CALLS  | #2, DBG\$PRINT          |      |
| 6B        |      | 01 | DD | 00544 |       | MOVL   | #1, DBG\$_PRINT_FLAG    | 2921 |
|           | 44   | AE | DD | 00547 |       | PUSHL  | REFTYPID                | 2922 |
| FAB1      | CF   | 01 | FB | 0054A | 52\$: | CALLS  | #1, SHOW_SYMBOL_TYPE    |      |
|           | 7E   | 04 | CE | 0054F |       | MNEGL  | #4, -(SP)               | 2923 |
|           |      | 02 | DD | 00552 |       | PUSHL  | #2                      |      |
| 6A        |      | 02 | FB | 00554 |       | CALLS  | #2, DBG\$PRINT_CONTROL  |      |
|           |      | 5D | 11 | 00557 | 53\$: | BRB    | 62\$                    | 2737 |
|           |      | 7E | D4 | 00559 | 54\$: | CLRL   | -(SP)                   | 2930 |
|           | FF75 | C7 | 9F | 0055B |       | PUSHAB | P.AIL                   |      |
|           |      | 52 | 11 | 0055F | 55\$: | BRB    | 61\$                    |      |
|           |      | 7E | D4 | 00561 | 56\$: | CLRL   | -(SP)                   | 2935 |
|           | 8A   | A7 | 9F | 00563 |       | PUSHAB | P.AIM                   |      |
|           |      | 4B | 11 | 00566 |       | BRB    | 61\$                    |      |
|           |      | 7E | D4 | 00568 | 57\$: | CLRL   | -(SP)                   | 2940 |
|           | 9E   | A7 | 9F | 0056A |       | PUSHAB | P.AIN                   |      |
|           |      | 44 | 11 | 0056D | 58\$: | BRB    | 61\$                    |      |
|           |      | 7E | D4 | 0056F | 59\$: | CLRL   | -(SP)                   | 2945 |
|           | B2   | A7 | 9F | 00571 |       | PUSHAB | P.AIO                   |      |
| 68        |      | 02 | FB | 00574 |       | CALLS  | #2, DBG\$PRINT          |      |
|           | 4C   | AE | 9F | 00577 |       | PUSHAB | AREA_LEN                | 2946 |
|           | 0C   | AE | DD | 0057A |       | PUSHL  | TYPEID                  |      |
| 00000000G | 00   | 02 | FB | 0057D |       | CALLS  | #2, DBG\$STA_TYP_AREA   |      |
|           | 4C   | AE | DD | 00584 |       | PUSHL  | AREA_LEN                | 2947 |
|           | BD   | A7 | 9F | 00587 |       | PUSHAB | P.AIP                   |      |
|           |      | 27 | 11 | 0058A |       | BRB    | 61\$                    |      |
|           |      | 7E | D4 | 0058C | 60\$: | CLRL   | -(SP)                   | 2952 |
|           | CE   | A7 | 9F | 0058E |       | PUSHAB | P.AIQ                   |      |
| 68        |      | 02 | FB | 00591 |       | CALLS  | #2, DBG\$PRINT          |      |
|           | 4C   | AE | 9F | 00594 |       | PUSHAB | AREA_LEN                | 2953 |
|           | 54   | AE | 9F | 00597 |       | PUSHAB | AREA_ADDR               |      |
|           | 10   | AE | DD | 0059A |       | PUSHL  | TYPEID                  |      |
| 00000000G | 00   | 03 | FB | 0059D |       | CALLS  | #3, DBG\$STA_TYP_OFFSET |      |



|           |    |    |      |       |        |                              |        |
|-----------|----|----|------|-------|--------|------------------------------|--------|
|           | 50 | AE | DD   | 005A4 | PUSHL  | AREA_ADDR                    | : 2954 |
|           | DB | A7 | 9F   | 005A7 | PUSHAB | P.AIR                        | : 2955 |
| 68        |    | 02 | FB   | 005AA | CALLS  | #2, DBG\$PRINT               | : 2965 |
|           | 4C | AE | DD   | 005AD | PUSHL  | AREA_LEN                     | : 2968 |
|           | EF | A7 | 9F   | 005B0 | PUSHAB | P.AIS                        | : 2969 |
| 68        |    | 02 | FB   | 005B3 | CALLS  | #2, DBG\$PRINT               | : 2975 |
| 05        |    | 6B | E9   | 005B6 | BLBC   | DBG\$ PRINT FLAG, 63\$       | : 2976 |
| 69        |    | 00 | FB   | 005B9 | CALLS  | #0, DBG\$NEWLINE             | : 2979 |
|           |    | 6B | D4   | 005BC | CLRL   | DBG\$ PRINT FLAG             | : 2683 |
| 00000000G | 00 | 00 | FB   | 005BE | CALLS  | #0, DBG\$RST_TEMP_RELEASE    |        |
|           |    | 56 | DD   | 005C5 | PUSHL  | POOLID                       |        |
| 00000000G | 00 | 01 | FB   | 005C7 | CALLS  | #1, DBG\$POP_TEMP_MEM        |        |
|           |    |    | 04   | 005CE | RET    |                              |        |
|           |    |    | 0000 | 005CF | .WORD  | Save nothing                 |        |
|           |    | 7E | D4   | 005D1 | CLRL   | -(SP)                        |        |
|           |    | 5E | DD   | 005D3 | PUSHL  | SP                           |        |
| 0000V     | 7E | 04 | AC   | 7D    | MOVQ   | 4(AP), -(SP)                 |        |
|           | CF |    | 03   | FB    | CALLS  | #3, SHOW_SYMBOL_TYPE_HANDLER |        |
|           |    |    | 04   | 005DE | RET    |                              |        |

; Routine Size: 1503 bytes, Routine Base: DBG\$CODE + 12F6



```

: 2863 2980 1 ROUTINE SHOW_SYMBOL_TYPE_HANDLER(SIGARG, MECHARG, ENBLARG) =
: 2864 2981 1
: 2865 2982 1 FUNCTION
: 2866 2983 1 This routine is the error handler for the SHOW_SYMBOL_TYPE routine.
: 2867 2984 1 It catches the Access Violations which occur during the evaluation
: 2868 2985 1 of DST Value Specs in DBG$STA_VALSPEC routine. It also catches the
: 2869 2986 1 "Symbol not in active scope" message during DST Value Spec evaluation
: 2870 2987 1 in DBG$STA_VALSPEC if a register is referenced which is not available
: 2871 2988 1 in the current context.
: 2872 2989 1
: 2873 2990 1 INPUTS
: 2874 2991 1 SIGARG - The signal argument vector.
: 2875 2992 1
: 2876 2993 1 MECHARG - The mechanism argument vector.
: 2877 2994 1
: 2878 2995 1 ENBLARG - The enable argument vector (not used here).
: 2879 2996 1
: 2880 2997 1 OUTPUT
: 2881 2998 1 For DBG$_ACCADDCOM or DBG$_SYMNOTACT error, we'll call DBG$PRINT
: 2882 2999 1 routine to print out message. For all other errors, this routine
: 2883 3000 1 just resignals.
: 2884 3001 1
: 2885 3002 1
: 2886 3003 2 BEGIN
: 2887 3004 2
: 2888 3005 2 MAP
: 2889 3006 2 SIGARG: REF VECTOR[,LONG]; ! Pointer to the signal argument vector
: 2890 3007 2
: 2891 3008 2 IF .SIGARG[1] NEQ DBG$_ACCADDCOM AND
: 2892 3009 2 .SIGARG[1] NEQ DBG$_SYMNOTACT
: 2893 3010 2 THEN
: 2894 3011 2 RETURN SSS$_RESIGNAL;
: 2895 3012 2
: 2896 3013 2 DBG$PRINT(UPLIT BYTE(%ASCIC ' (type/address information is not available)'), 0);
: 2897 3014 2 DBG$NEWLINE();
: 2898 3015 2 SETUNWIND();
: 2899 3016 2 RETURN 0;
: 2900 3017 2
: 2901 3018 1 END;

```

```

: 73 73 65 72 64 64 61 2F 65 70 79 74 28 20 2C 00C8C P.AIU: .ASCII \, (type/address information is not avail\
: 73 69 20 6E 6F 69 74 61 6D 72 6F 66 6E 69 20 00C9B
: 6C 69 61 76 61 20 74 6F 6E 20 00CAA
: 29 65 6C 62 61 00CB4 .ASCII \able)\

```

.PSECT DBG\$CODE,NOWRT, SHR, PIC,0

0000 00000 SHOW\_SYMBOL TYPE HANDLER:

```

00028C98 50 04 AC D0 00002 .WORD Save nothing
8F 04 A0 D1 00006 MOVL SIGARG, R0
CMPL 4(R0), #167064

```

: 2980  
: 3008  
:



DBGSYMBOL  
V04-000

E 4  
16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

VAX-11 Bliss-32 V4.0-742  
[DEBUG.SRC]DBGSYMBOL.B32;1

Page 106  
(17)

|           |    |           |    |    |       |        |                  |   |      |
|-----------|----|-----------|----|----|-------|--------|------------------|---|------|
| 00028C88  | 8F | 04        | 10 | 13 | 0000E | BEQL   | 1\$              | : |      |
|           |    |           | A0 | D1 | 00010 | CMPL   | 4(R0), #167048   | : | 3009 |
|           | 50 | 0918      | 06 | 13 | 00018 | BEQL   | 1\$              | : |      |
|           |    |           | 8F | 3C | 0001A | MOVZWL | #2328, R0        | : | 3011 |
|           |    |           |    | 04 | 0001F | RET    |                  | : |      |
|           |    |           | 7E | D4 | 00020 | CLRL   | -(SP)            | : | 3013 |
|           |    | 00000000' | EF | 9F | 00022 | PUSHAB | P, AIU           | : |      |
| 00000000G | 00 |           | 02 | FB | 00028 | CALLS  | #2, DBG\$PRINT   | : |      |
| 00000000G | 00 |           | 00 | FB | 0002F | CALLS  | #0, DBG\$NEWLINE | : | 3014 |
|           |    |           | 7E | 7C | 00036 | CLRQ   | -(SP)            | : | 3015 |
| 00000000G | 00 |           | 02 | FB | 00038 | CALLS  | #2, SYSSUNWIND   | : |      |
|           |    |           | 50 | D4 | 0003F | CLRL   | R0               | : | 3016 |
|           |    |           |    | 04 | 00041 | RET    |                  | : | 3018 |

; Routine Size: 66 bytes,      Routine Base: DBG\$CODE + 18D5



```
2903 3019 1 ROUTINE WILDCARD_CANDIDATES(SYMBOL_PTR, MODRSTPTR, RSTPTR): =
2904 3020 1
2905 3021 1 FUNCTION
2906 3022 1 This routine follows the symbol chain pointer in all set modules, and
2907 3023 1 all global symbols pointed by RST$START_ADDR's symbol chain pointer.
2908 3024 1 For each symbol's RST entry, this routine gets the name of the
2909 3025 1 symbol and tries to match this name with the wildcard string pattern.
2910 3026 1 If the result matches, return RST entry pointer of this symbol.
2911 3027 1 This routine also updates the module RST pointer to point to the next
2912 3028 1 set module, and the symbol RST pointer to point to the next symbol
2913 3029 1 in the chain for the same module. 0 is returned when reached to
2914 3030 1 the end of the chain of all set modules.
2915 3031 1
2916 3032 1 INPUTS
2917 3033 1 SYMBOL_PTR - Pointer points to the wildcard name string after the
2918 3034 1 modification. Such as, pointer value is 0 if the wildcard
2919 3035 1 name string is "*", or pointer points to Sub-Strings, ie.
2920 3036 1 A*B*C -> ^0A^B^C0, *AB -> ^A^B0, AB* -> ^0A^B, etc.
2921 3037 1
2922 3038 1 MODRSTPTR - Pointer points to next set module RST entry
2923 3039 1
2924 3040 1 RSTPTR - Pointer points to next symbol RST entry belong to
2925 3041 1 the same module
2926 3042 1
2927 3043 1 OUTPUTS
2928 3044 1 This routine returns candidate symbol's RST entry, when we have
2929 3045 1 found the match, or 0 when reaches to end.
2930 3046 1
2931 3047 2 BEGIN
2932 3048 2
2933 3049 2 MAP
2934 3050 2 SYMBOL_PTR: REF WILDCARDS$_FLDS, ! Pointer to Wildcard Sub-String or 0
2935 3051 2 MODRSTPTR: REF VECTOR[1], ! Pointer to next set module's RST entry
2936 3052 2 RSTPTR: REF VECTOR[1]; ! Pointer to next symbol in symbol chain
2937 3053 2 ! belong to the same module
2938 3054 2
2939 3055 2 LOCAL
2940 3056 2 FND, ! Flag set to indicate we have found
2941 3057 2 ! the desired candidate symbol
2942 3058 2 FNDPTR: REF VECTOR[.BYTE], ! Pointer points to the matched Sub-
2943 3059 2 ! String in candidate symbol name
2944 3060 2 CAND_RSTPTR: REF RST$ENTRY, ! Pointer points to candidate symbol
2945 3061 2 ! RST entry
2946 3062 2 LENGTH, ! The remaining search length of the
2947 3063 2 ! candidate symbol name string
2948 3064 2 MATCH_NAME: REF VECTOR[.BYTE], ! Candidate symbol name appended with
2949 3065 2 ! begin and end characters
2950 3066 2 NAME: REF VECTOR[.BYTE], ! Candidate symbol name
2951 3067 2 NEXTSETMOD: REF RST$ENTRY, ! Pointer points to next set module
2952 3068 2 SUBSTR: REF VECTOR[.BYTE], ! Pointer to sub-string in wildcard
2953 3069 2 ! sub-string link list
2954 3070 2 WILDCARD_PTR: REF WILDCARDS$_FLDS; ! Pointer to wildcard sub-string link
2955 3071 2 ! list
2956 3072 2
2957 3073 2
2958 3074 2 MATCH_NAME = 0;
2959 3075 2
```



```
2960 3076 2
2961 3077
2962 3078
2963 3079
2964 3080
2965 3081
2966 3082
2967 3083
2968 3084
2969 3085
2970 3086
2971 3087
2972 3088
2973 3089
2974 3090
2975 3091
2976 3092
2977 3093
2978 3094
2979 3095
2980 3096
2981 3097
2982 3098
2983 3099
2984 3100
2985 3101
2986 3102
2987 3103
2988 3104
2989 3105
2990 3106
2991 3107
2992 3108
2993 3109
2994 3110
2995 3111
2996 3112
2997 3113
2998 3114
2999 3115
3000 3116
3001 3117
3002 3118
3003 3119
3004 3120
3005 3121
3006 3122
3007 3123
3008 3124
3009 3125
3010 3126
3011 3127
3012 3128
3013 3129
3014 3130
3015 3131
3016 3132

! Adjust the pointers to start matching point in symbol chain of set
! module. Assume we have not found the candidate.
NEXTSETMOD = .MODRSTPTR[0];
CAND_RSTPTR = .RSTPTR[0];
FND = FALSE;

! Walk through the symbol chain in all set modules and try to match
! the candidate symbol name with the desired wildcard string pattern.
WHILE TRUE DO
  BEGIN

    ! Check to see if we have reached to the end of the global symbol
    ! chain, if we have, then we are all done.
    IF .CAND_RSTPTR EQL 0 AND .NEXTSETMOD EQL 0
    THEN
      BEGIN
        IF .MATCH_NAME NEQ 0 THEN DBGSREL_MEMORY(.MATCH_NAME);
        RETURN .CAND_RSTPTR;
      END;

    ! Check to see if we have reached to the end of the symbol chain
    ! belong to the same module. If we have, set the module RST
    ! entry pointer to next set module. If we have gone through all
    ! the set modules, return 0 to indicate the case.
    IF .CAND_RSTPTR EQL 0 AND .NEXTSETMOD NEQ 0
    THEN
      BEGIN

        ! Set the module pointer to next set module.
        WHILE .NEXTSETMOD NEQ 0 DO
          BEGIN

            ! Set the pointer to next module in module chain.
            NEXTSETMOD = .NEXTSETMOD[RST$LNXTMODPTR];

            ! Check to see if we have reached to the end of all set modules.
            ! If we have, exit this loop.
            IF .NEXTSETMOD EQL 0 THEN EXITLOOP;

            ! Check to see if this module is set. If it is, exit, then
            ! walk through its symbol chain.
```



```

3017      3133 5      IF .NEXTSETMOD[RST$V_MODSET] THEN EXITLOOP;
3018      3134 4      END;
3019      3135 4      ! End of WHILE setting module pointer.
3020      3136 4
3021      3137 4      ! Set the candidate symbol chain pointer to start point.
3022      3138 4      ! If we have reached to the end of all set modules, set the pointer
3023      3139 4      ! to search global symbols.
3024      3140 4
3025      3141 4      IF .NEXTSETMOD NEQ 0
3026      3142 4      THEN
3027      3143 4          CAND_RSTPTR = .NEXTSETMOD
3028      3144 4      ELSE
3029      3145 4          CAND_RSTPTR = .RST$START_ADDR[RST$L_SYMCHNPTR];
3030      3146 4
3031      3147 4      END;
3032      3148 4
3033      3149 4
3034      3150 4      ! If the value of SYMBOL_PTR is zero implies we want to show every
3035      3151 4      ! symbol in the DEBUG Run-Time Symbol. So simply updates symbol
3036      3152 4      ! RST pointer to next symbol in the chain, and updates the module
3037      3153 4      ! RST pointer to next set module at the end of the symbol chain,
3038      3154 4      ! then return. For the other case, we have a wildcard string pattern
3039      3155 4      ! to match.
3040      3156 4
3041      3157 4      IF .SYMBOL_PTR NEQ 0
3042      3158 4      THEN
3043      3159 4          BEGIN
3044      3160 4
3045      3161 4          ! Pick up candidate symbol's name.
3046      3162 4          !
3047      3163 4          NAME = DBG$GET_DST_NAME(.CAND_RSTPTR[RST$L_DSTPTR]);
3048      3164 4
3049      3165 4          ! Append begin and end characters to this candidate symbol's name.
3050      3166 4          !
3051      3167 4          MATCH_NAME = WILDCARD_NAME_SETUP(.NAME);
3052      3168 4
3053      3169 4          ! Match every Sub-String in wildcard sub-string linked list with
3054      3170 4          ! the candidate symbol name orderly. If we can match all Sub-String
3055      3171 4          ! patterns in candidate symbol name then we have found the desired
3056      3172 4          ! candidate.
3057      3173 4          !
3058      3174 4          FNDPTR = MATCH_NAME[1];
3059      3175 4          LENGTH = .MATCH_NAME[0];
3060      3176 4          WILDCARD_PTR = .SYMBOL_PTR;
3061      3177 4          WHILE TRUE DO
3062      3178 4              BEGIN
3063      3179 4
3064      3180 4              ! Check to see if we have reached to the end of the linked
3065      3181 5              ! list. If we have, at this point we have successfully found
3066      3182 5              ! the candidate.
3067      3183 5              !
3068      3184 5              IF .WILDCARD_PTR EQL 0
3069      3185 5              THEN
3070      3186 5
3071      3187 5
3072      3188 5
3073      3189 5
```



```
3074      3190      6      BEGIN
3075      3191      6      FND = TRUE;
3076      3192      6      EXITLOOP;
3077      3193      6      END;
3078      3194      6
3079      3195      6
3080      3196      6      ! Locate the Sub-String in the candidate symbol name string.
3081      3197      6
3082      3198      6      SUBSTR = .WILDCARD_PTR[WILDCARD$$_STRPTR];
3083      3199      6      FNDPTR = CH$FIND_SUB(.LENGTH, .FNDPTR, .SUBSTR[0], SUBSTR[1]);
3084      3200      6
3085      3201      6
3086      3202      6      ! If we failed to locate the Sub-String, no point to go on.
3087      3203      6
3088      3204      6      IF .FNDPTR EQL 0 THEN EXITLOOP;
3089      3205      6
3090      3206      6
3091      3207      6      ! Pick up next Sub-String in the wildcard sub-string linked
3092      3208      6      list.
3093      3209      6
3094      3210      6      WILDCARD_PTR = .WILDCARD_PTR[WILDCARD$$_LINK];
3095      3211      6
3096      3212      6
3097      3213      6      ! Set the pointer and length to the remaining candidate symbol
3098      3214      6      name string.
3099      3215      6
3100      3216      6      FNDPTR = .FNDPTR + .SUBSTR[0];
3101      3217      6      LENGTH = .LENGTH - .SUBSTR[0];
3102      3218      6      END;
3103      3219      6
3104      3220      6      END
3105      3221      6
3106      3222      6
3107      3223      6      ! Every symbol in RST is a candidate. Set the flag to indicate we
3108      3224      6      have a candidate.
3109      3225      6
3110      3226      6      ELSE
3111      3227      6          FND = TRUE;
3112      3228      6
3113      3229      6
3114      3230      6      ! Check to see if we have found the desired candidate symbol. If
3115      3231      6      we have, exit from walking RST entries.
3116      3232      6
3117      3233      6      IF .FND THEN EXITLOOP;
3118      3234      6
3119      3235      6
3120      3236      6      ! We have not found the candidate symbol yet, continue to next
3121      3237      6      candidate symbol in RST entries.
3122      3238      6
3123      3239      6      CAND_RSTPTR = .CAND_RSTPTR[RST$$_SYMCHNPTR];
3124      3240      6      IF .MATCH_NAME NEQ 0
3125      3241      6      THEN
3126      3242      6          BEGIN
3127      3243      6              DBG$REL_MEMORY(.MATCH_NAME);
3128      3244      6              MATCH_NAME = 0;
3129      3245      6          END;
3130      3246      6
```



3131  
3132  
3133  
3134  
3135  
3136  
3137  
3138  
3139  
3140  
3141  
3142

3247  
3248  
3249  
3250  
3251  
3252  
3253  
3254  
3255  
3256  
3257  
3258

2  
2  
2  
2  
2  
2  
2  
2  
2  
2  
2  
1

END;

! End of WHILE walking the RST entries.

! Update the Module RST pointer and Symbol RST pointer. Returned selected  
candidate symbol.

MODRSTPTR[0] = .NEXTSETMOD;  
RSTPTR[0] = .CAND\_RSTPTR[RST\$SYMCHNPTR];  
IF .MATCH\_NAME NEQ 0 THEN DBGSREL\_MEMORY(.MATCH\_NAME);  
RETURN .CAND\_RSTPTR;

END;

OFFC 00000 WILDCARD\_CANDIDATES:

|           |           |      |    |       |        |                                      |      |
|-----------|-----------|------|----|-------|--------|--------------------------------------|------|
| 5E        |           | 04   | C2 | 00002 | WORD   | Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 | 3019 |
| 54        | 08        | BC   | D0 | 00005 | SUBL2  | #4, SP                               | 3080 |
| 55        | 0C        | BC   | D0 | 00009 | MOVL   | @MODRSTPTR, NEXTSETMOD               | 3081 |
|           |           | 59   | 7C | 0000D | MOVL   | @RSTPTR, CAND_RSTPTR                 | 3074 |
|           |           | 52   | D4 | 0000F | CLRQ   | MATCH_NAME                           | 3095 |
|           |           | 55   | D5 | 00011 | CLRL   | R2                                   |      |
|           |           | 10   | 12 | 00013 | TSTL   | CAND_RSTPTR                          |      |
|           |           | 52   | D6 | 00015 | BNEQ   | 3\$                                  |      |
|           |           | 54   | D5 | 00017 | INCL   | R2                                   |      |
|           |           | 0A   | 12 | 00019 | TSTL   | NEXTSETMOD                           |      |
|           |           | 59   | D5 | 0001B | BNEQ   | 3\$                                  |      |
|           |           | 03   | 12 | 0001D | TSTL   | MATCH_NAME                           | 3098 |
|           |           | 00AD | 31 | 0001F | BNEQ   | 2\$                                  |      |
|           |           | 00A1 | 31 | 00022 | BRW    | 14\$                                 |      |
| 22        |           | 52   | E9 | 00025 | BRW    | 13\$                                 |      |
|           |           | 54   | D5 | 00028 | BLBC   | R2, 6\$                              | 3108 |
|           |           | 1E   | 13 | 0002A | TSTL   | NEXTSETMOD                           |      |
|           |           | 11   | 13 | 0002C | BEQL   | 6\$                                  |      |
| 54        | 10        | A4   | D0 | 0002E | BEQL   | 5\$                                  | 3115 |
|           |           | 0B   | 13 | 00032 | MOVL   | 16(NEXTSETMOD), NEXTSETMOD           | 3121 |
| F4        | 28        | A4   | E9 | 00034 | BEQL   | 5\$                                  | 3127 |
|           |           | 05   | 13 | 00038 | BLBC   | 40(NEXTSETMOD), 4\$                  | 3133 |
| 55        |           | 54   | D0 | 0003A | BEQL   | 5\$                                  | 3141 |
|           |           | 0B   | 11 | 0003D | MOVL   | NEXTSETMOD, CAND_RSTPTR              | 3143 |
| 50        | 00000000G | 00   | D0 | 0003F | BRB    | 6\$                                  |      |
| 55        |           | 08   | A0 | 00046 | MOVL   | RST\$START_ADDR, R0                  | 3145 |
|           |           | 04   | AC | D5    | MOVL   | 8(R0), CAND_RSTPTR                   |      |
|           |           | 4E   | 13 | 0004D | TSTL   | SYMBOL_PTR                           | 3157 |
|           |           | 0C   | A5 | DD    | BEQL   | 9\$                                  |      |
| 00000000G | 00        | 01   | FB | 00052 | PUSHL  | 12(CAND_RSTPTR)                      | 3164 |
|           | 6E        | 50   | D0 | 00059 | CALLS  | #1, DBGSGET_DST_NAME                 |      |
|           |           | 6E   | DD | 0005C | MOVL   | R0, NAME                             |      |
| 0000V     | CF        | 01   | FB | 0005E | PUSHL  | NAME                                 | 3169 |
|           | 59        | 50   | D0 | 00063 | CALLS  | #1, WILDCARD_NAME_SETUP              |      |
|           | 53        | 01   | A9 | 9E    | MOVL   | R0, MATCH_NAME                       | 3177 |
|           | 5B        | 69   | 9A | 0006A | MOVAB  | 1(R9), FNDPTR                        |      |
|           | 56        | 04   | AC | D0    | MOVZBL | (MATCH_NAME), LENGTH                 | 3178 |
|           |           | 56   | D5 | 00071 | MOVL   | SYMBOL_PTR, WILDCARD_PTR             | 3179 |
|           |           |      |    |       | TSTL   | WILDCARD_PTR                         | 3188 |



|    |    |           |    |    |      |       |       |        |                                 |                             |      |
|----|----|-----------|----|----|------|-------|-------|--------|---------------------------------|-----------------------------|------|
| 63 | 5B | 01        | 57 | 04 | 28   | 13    | 00073 | BEQL   | 9\$                             |                             |      |
|    |    |           | 58 |    | A6   | D0    | 00075 | MOVL   | 4(WILDCARD_PTR), SUBSTR         | ...                         | 3198 |
|    |    |           | A7 |    | 67   | 9A    | 00079 | MOVZBL | (SUBSTR), R8                    | ...                         | 3199 |
|    |    |           |    |    | 58   | 39    | 0007C | MATCHC | R8, 1(SUBSTR), LENGTH, (FNDPTR) | ...                         |      |
|    |    |           | 53 |    | 03   | 13    | 00082 | BEQL   | 8\$                             | ...                         |      |
|    |    |           | 53 |    | 58   | D0    | 00084 | MOVL   | R8, R3                          | ...                         |      |
|    |    |           |    |    | 58   | C2    | 00087 | SUBL2  | R8, R3                          | ...                         |      |
|    |    |           | 56 |    | 14   | 13    | 0008A | BEQL   | 10\$                            | ...                         | 3204 |
|    |    |           | 50 |    | 66   | D0    | 0008C | MOVL   | (WILDCARD_PTR), WILDCARD_PTR    | ...                         | 3210 |
|    |    |           | 53 |    | 67   | 9A    | 0008F | MOVZBL | (SUBSTR), R0                    | ...                         | 3216 |
|    |    |           | 50 |    | 50   | C0    | 00092 | ADDL2  | R0, FNDPTR                      | ...                         |      |
|    |    |           | 50 |    | 67   | 9A    | 00095 | MOVZBL | (SUBSTR), R0                    | ...                         | 3217 |
|    |    |           | 5B |    | 50   | C2    | 00098 | SUBL2  | R0, LENGTH                      | ...                         |      |
|    |    |           |    |    | D4   | 11    | 0009B | BRB    | 7\$                             | ...                         | 3180 |
|    |    |           | 5A |    | 01   | D0    | 0009D | MOVL   | #1, FND                         | ...                         | 3227 |
|    |    |           | 16 |    | 5A   | E8    | 000A0 | BLBS   | FND, 12\$                       | ...                         | 3233 |
|    |    |           | 55 |    | 08   | A5    | D0    | 000A3  | MOVL                            | 8(CAND_RSTPTR), CAND_RSTPTR | 3239 |
|    |    |           |    |    | 59   | D5    | 000A7 | TSTL   | MATCH_NAME                      | ...                         | 3240 |
|    |    |           |    |    | 0B   | 13    | 000A9 | BEQL   | 11\$                            | ...                         |      |
|    |    |           |    |    | 59   | DD    | 000AB | PUSHL  | MATCH_NAME                      | ...                         | 3243 |
|    |    | 00000000G | 00 |    | 01   | FB    | 000AD | CALLS  | #1, DBG\$REL_MEMORY             | ...                         |      |
|    |    |           |    |    | 59   | D4    | 000B4 | CLRL   | MATCH_NAME                      | ...                         | 3244 |
|    |    |           |    |    | FF56 | 31    | 000B6 | BRW    | 1\$                             | ...                         | 3088 |
|    |    | 08        | BC |    | 54   | D0    | 000B9 | MOVL   | NEXTSETMOD, @MODRSTPTR          | ...                         | 3253 |
|    |    | 0C        | BC |    | 08   | A5    | D0    | 000BD  | MOVL                            | 8(CAND_RSTPTR), @RSTPTR     | 3254 |
|    |    |           |    |    | 59   | D5    | 000C2 | TSTL   | MATCH_NAME                      | ...                         | 3255 |
|    |    |           |    |    | 09   | 13    | 000C4 | BEQL   | 14\$                            | ...                         |      |
|    |    |           |    |    | 59   | DD    | 000C6 | PUSHL  | MATCH_NAME                      | ...                         |      |
|    |    | 00000000G | 00 |    | 01   | FB    | 000C8 | CALLS  | #1, DBG\$REL_MEMORY             | ...                         |      |
|    |    |           | 50 |    | 55   | D0    | 000CF | MOVL   | CAND_RSTPTR, R0                 | ...                         | 3256 |
|    |    |           |    |    | 04   | 000D2 | RET   |        |                                 | ...                         | 3258 |

; Routine Size: 211 bytes, Routine Base: DBG\$CODE + 1917

; 3143 3259 1



```
3145 3260 1 ROUTINE WILDCARD_NAME_SETUP(SYMBOL_NAME): =
3146 3261 1
3147 3262 1 FUNCTION
3148 3263 1     In order to make the matching wildcard format correctly and simply,
3149 3264 1     this routine appends null characters at the begin and end string if
3150 3265 1     the begin and end characters are not '*'.
3151 3266 1
3152 3267 1 INPUTS
3153 3268 1     SYMBOL_NAME - Counted ASCII name string
3154 3269 1
3155 3270 1 OUTPUTS
3156 3271 1     Return Counted ASCII name string appended with null characters or
3157 3272 1     Counted ASCII name string itself.
3158 3273 1
3159 3274 1
3160 3275 2 BEGIN
3161 3276 2
3162 3277 2 MAP
3163 3278 2     SYMBOL_NAME: REF VECTOR[BYTE]; ! Counted ASCII name string
3164 3279 2
3165 3280 2 LOCAL
3166 3281 2     IDX, ! Indexed into ith byte in the string
3167 3282 2     LEFT_FLAG, ! Flag set to indicate we need to pad
3168 3283 2     ! null character at the begin
3169 3284 2     ! string
3170 3285 2     LENGTH, ! Length of the counted ASCII string
3171 3286 2     RIGHT_FLAG, ! Flag set to indicate we need to pad
3172 3287 2     ! null character at the end string
3173 3288 2     STRING: REF VECTOR[BYTE]; ! Counted ASCII string after modification
3174 3289 2
3175 3290 2
3176 3291 2 ! Assume we do not need the paddings.
3177 3292 2
3178 3293 2 LEFT_FLAG = FALSE;
3179 3294 2 RIGHT_FLAG = FALSE;
3180 3295 2 LENGTH = .SYMBOL_NAME[0];
3181 3296 2
3182 3297 2
3183 3298 2 ! Check begin character.
3184 3299 2
3185 3300 2 IF .SYMBOL_NAME[1] NEQ %C'*'
3186 3301 2 THEN
3187 3302 2     BEGIN
3188 3303 2     LENGTH = .LENGTH + 1;
3189 3304 2     LEFT_FLAG = TRUE;
3190 3305 2     END;
3191 3306 2
3192 3307 2
3193 3308 2 ! Check end character.
3194 3309 2
3195 3310 2 IF .SYMBOL_NAME[.SYMBOL_NAME[0]] NEQ %C'*'
3196 3311 2 THEN
3197 3312 2     BEGIN
3198 3313 2     LENGTH = .LENGTH + 1;
3199 3314 2     RIGHT_FLAG = TRUE;
3200 3315 2     END;
3201 3316 2
```



```

3202      3317 2
3203      3318
3204      3319
3205      3320
3206      3321
3207      3322
3208      3323
3209      3324
3210      3325
3211      3326
3212      3327
3213      3328
3214      3329
3215      3330
3216      3331
3217      3332
3218      3333
3219      3334
3220      3335
3221      3336
3222      3337
3223      3338
3224      3339
3225      3340
3226      3341
3227      3342
3228      3343
3229      3344
3230      3345
3231      3346
3232      3347
3233      3348
3234      3349 1

```

```

! Get some memory for modified string.
STRING = DBG$GET_MEMORY(.LENGTH/4 + 1);

! Pad null character if the flag is set.
IDX = 1;
IF .LEFT_FLAG
THEN
  BEGIN
    STRING[1] = 0;
    IDX = .IDX + 1;
  END;

! Copy the string itself.
CH$MOVE(.SYMBOL_NAME[0], SYMBOL_NAME[1], STRING[.IDX]);

! Pad null character if the flag is set.
IF .RIGHT_FLAG THEN STRING[.LENGTH] = 0;

! Adjust the length of the string and return the modified string.
STRING[0] = .LENGTH;
RETURN .STRING;

END;

```

```

01FC 0000 WILDCARD_NAME_SETUP:
      53 D4 00002      WORD Save R2,R3,R4,R5,R6,R7,R8
      58 D4 00004      CLRL LEFT_FLAG
      52 04 AC D0 00006      CLRL RIGHT_FLAG
      56 62 9A 0000A      MOVL SYMBOL_NAME, R2
      2A 01 A2 91 0000D      MOVZBL (R2), LENGTH
      05 13 00011      CMPB 1(R2), #42
      56 D6 00013      BEQL 1$
      01 D0 00015      INCL LENGTH
      50 62 9A 00018 1$:      MOVL #1, LEFT_FLAG
      2A 62 9A 0001B      MOVZBL (R2), R0
      05 13 0001F      CMPB (R2)(R0), #42
      56 D6 00021      BEQL 2$
      01 D0 00023      INCL LENGTH
      58 04 C7 00026 2$:      MOVL #1, RIGHT_FLAG
      50 01 A0 9F 0002A      DIVL3 #4, LENGTH, R0
      00 01 FB 0002D      PUSHAB 1(R0)
      57 50 D0 00034      CALLS #1, DBG$GET_MEMORY
      50 D0 00034      MOVL R0, STRING

```



DBGSYMBOL  
V04-000

N 4  
16-Sep-1984 02:39:38  
14-Sep-1984 12:17:52

VAX-11 Bliss-32 V4.0-742  
[DEBUG.SRC]DBGSYMBOL.B32;1

Page 115  
(19)

|      |    |    |      |    |       |        |                            |        |
|------|----|----|------|----|-------|--------|----------------------------|--------|
|      |    | 51 | 01   | D0 | 00037 | MOVL   | #1, IDX                    | : 3325 |
|      |    | 05 | 53   | E9 | 0003A | BLBC   | LEFT FLAG, 3\$             | : 3326 |
|      |    |    | A7   | 94 | 0003D | CLRB   | 1 (STRING)                 | : 3329 |
|      |    |    | 51   | D6 | 00040 | INCL   | IDX                        | : 3330 |
|      |    | 50 | 62   | 9A | 00042 | MOVZBL | (R2), R0                   | : 3336 |
| 6147 | 01 | A2 | 50   | 28 | 00045 | MOV3   | R0, 1 (R2), (IDX) [STRING] | : 3341 |
|      |    | 03 | 58   | E9 | 0004B | BLBC   | RIGHT FLAG, 4\$            | : 3346 |
|      |    |    | 6647 | 94 | 0004E | CLRB   | (LENGTH) [STRING]          | : 3347 |
|      |    | 67 | 56   | 90 | 00051 | MOVB   | LENGTH, (STRING)           | : 3349 |
|      |    | 50 | 57   | D0 | 00054 | MOVL   | STRING, R0                 | : 3349 |
|      |    |    |      | 04 | 00057 | RET    |                            |        |

; Routine Size: 88 bytes, Routine Base: DBG\$CODE + 19EA



```
3236 1 ROUTINE WILDCARD_SUB_STRING(SYMBOL_NAME): =
3237 1
3238 1 FUNCTION
3239 1     This routine breaks wildcard name string into Sub-Strings.
3240 1
3241 1 INPUTS
3242 1     SYMBOL_NAME - Counted ASCII string contains wildcard characters.
3243 1
3244 1 OUTPUTS
3245 1     Return a linked list contains Sub-Strings, or zero if '*' is the
3246 1     wildcard name string.
3247 1
3248 1 BEGIN
3249 1
3250 1 MAP
3251 1     SYMBOL_NAME: REF VECTOR[BYTE]; ! Counted ASCII wildcard name string
3252 1
3253 1 LOCAL
3254 1     LENGTH, ! Remaining length of the wildcard
3255 1             ! name string
3256 1     NAMEPTR: REF VECTOR[BYTE], ! Pointer to Sub-String
3257 1     START: REF VECTOR[BYTE], ! Pointer to counted ASCII wildcard
3258 1             ! name string
3259 1     SUB_STRING: REF WILDCARD$_FLDS, ! Pointer to linked list
3260 1     SYMBOL_PTR: REF WILDCARD$_FLDS, ! Pointer to linked list
3261 1     WILDCARD_PTR: REF VECTOR[BYTE]; ! Pointer to '*' in the wildcard name
3262 1             ! string
3263 1
3264 1
3265 1 SYMBOL_PTR = 0;
3266 1
3267 1 ! Wildcard name string is '*', no need to break it into Sub-Strings.
3268 1
3269 1 IF .SYMBOL_NAME[0] EQL 1 THEN RETURN .SYMBOL_PTR;
3270 1
3271 1 ! Creat linked list node.
3272 1
3273 1 SYMBOL_PTR = DBG$GET_TEMPMEM(WILDCARD$_SIZE);
3274 1 SUB_STRING = .SYMBOL_PTR;
3275 1 LENGTH = .SYMBOL_NAME[0];
3276 1 START = SYMBOL_NAME[1];
3277 1
3278 1
3279 1 ! Locate the '*' in the wildcard name string, extract the Sub-String
3280 1 ! without the '*', insert it onto the linked list.
3281 1
3282 1 WHILE TRUE DO
3283 1     BEGIN
3284 1         WILDCARD_PTR = CH$FIND_CH(.LENGTH, .START, %C'*');
3285 1
3286 1         ! Check for end case. If WILDCARD_PTR is zero, this implies no more
3287 1         ! '*' can be found in the wildcard name string. Set WILDCARD_PTR
3288 1         ! points to the end of wildcard name string.
3289 1
3290 1
3291 1
3292 1
```



```

3293 3407 3
3294 3408
3295 3409
3296 3410
3297 3411
3298 3412
3299 3413
3300 3414
3301 3415
3302 3416
3303 3417
3304 3418
3305 3419
3306 3420
3307 3421
3308 3422
3309 3423
3310 3424
3311 3425
3312 3426
3313 3427
3314 3428
3315 3429
3316 3430
3317 3431
3318 3432
3319 3433
3320 3434
3321 3435
3322 3436
3323 3437
3324 3438
3325 3439
3326 3440
3327 3441
3328 3442
3329 3443
3330 3444
3331 3445
3332 3446 1

IF .WILDCARD_PTR EQL 0 THEN WILDCARD_PTR = SYMBOL_NAME[1] + .SYMBOL_NAME[0];

! Get some tempory memory for Sub-String.
NAMEPTR = DBG$GET TEMPMEM((.WILDCARD_PTR - .START)/ 4 + 1);
NAMEPTR[0] = .WILDCARD_PTR - .START;

! Extract the Sub-String from wildcard name string. Install Sub-string
! pointer onto linked list.
CH$MOVE(.NAMEPTR[0], .START, NAMEPTR[1]);
SUB_STRING[WILDCARD$L_STRPTR] = .NAMEPTR;

! Check to see if we have reached to the end of the string. If we
! have, then we are all done.
IF .WILDCARD_PTR GEQ (SYMBOL_NAME[1] + .SYMBOL_NAME[0] - 1) THEN EXITLOOP;

! More Sub-String to break, get more tempory memory for next node.
SUB_STRING[WILDCARD$L_LINK] = DBG$GET TEMPMEM(WILDCARD$K_SIZE);
SUB_STRING = .SUB_STRING[WILDCARD$L_LINK];

! Adjust the pointer and remaining length.
START = .WILDCARD_PTR + 1;
LENGTH = SYMBOL_NAME[1] + .SYMBOL_NAME[0] - .START;
END;

! End of the linked list. Return linked list pointer.
SUB_STRING[WILDCARD$L_LINK] = 0;
RETURN .SYMBOL_PTR;
END;
```

```

OFFC 00000 WILDCARD SUB STRING:
                                -WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11
                                CLRL SYMBOL_PTR
                                CMPB @SYMBOL_NAME, #1
                                BNEQ 1$
                                BRW 6$
                                PUSHL #2
                                CALLS #1, DBG$GET TEMPMEM
                                MOVL R0, SYMBOL_PTR
                                MOVL SYMBOL_PTR, SUB_STRING
                                MOVZBL @SYMBOL_NAME, LENGTH
                                ADDL3 #1, SYMBOL_NAME, START
                                : 3350
                                : 3379
                                : 3384
                                : 3389
                                : 3390
                                : 3391
                                : 3392

01 04 5B D4 00002
BC 91 00004
03 12 00008
0081 31 0000A 1$:
02 DD 0000D
01 FB 0000F
5B 50 D0 00016
57 5B D0 00019
56 04 BC 9A 0001C
59 04 AC 01 C1 00020
```



|    |           |    |    |       |       |        |                         |                          |
|----|-----------|----|----|-------|-------|--------|-------------------------|--------------------------|
| 69 | 56        | 2A | 3A | 00025 | 2\$:  | LOCC   | #42, LENGTH, (START)    | 3400                     |
|    |           | 02 | 12 | 00029 |       | BNEQ   | 3\$                     |                          |
|    |           | 51 | D4 | 0002B |       | CLRL   | R1                      |                          |
|    | 5A        | 51 | D0 | 0002D | 3\$:  | MOVL   | R1, WILDCARD_PTR        |                          |
|    |           | 0C | 12 | 00030 |       | BNEQ   | 4\$                     | 3407                     |
|    | 51        | 04 | BC | 9A    | 00032 | MOVZBL | @SYMBOL_NAME, R1        |                          |
|    | 51        | 04 | AC | C0    | 00036 | ADDL2  | SYMBOL_NAME, R1         |                          |
|    | 5A        | 01 | A1 | 9E    | 0003A | MOVAB  | 1(R1), WILDCARD_PTR     |                          |
| 52 | 5A        |    | 59 | C3    | 0003E | 4\$:   | SUBL3                   | START, WILDCARD_PTR, R2  |
| 51 | 52        |    | 04 | C7    | 00042 | DIVL3  | #4, R2, R1              | 3412                     |
|    |           | 01 | A1 | 9F    | 00046 | PUSHAB | 1(R1)                   |                          |
|    | 00000000G |    | 01 | FB    | 00049 | CALLS  | #1, DBG\$GET_TEMPMEM    |                          |
|    |           |    | 50 | D0    | 00050 | MOVL   | R0, NAMEPTR             |                          |
|    |           |    | 52 | 90    | 00053 | MOVB   | R2, (NAMEPTR)           | 3413                     |
|    |           |    | 68 | 9A    | 00056 | MOVZBL | (NAMEPTR), R0           | 3419                     |
| 01 | A8        |    | 50 | 28    | 00059 | MOVCL3 | R0, (START), 1(NAMEPTR) |                          |
|    |           |    | 58 | D0    | 0005E | MOVL   | NAMEPTR, 4(SUB_STRING)  | 3420                     |
|    | 04        |    | 52 | 04    | BC    | 9A     | 00062                   | MOVZBL                   |
|    |           |    | 52 | 04    | AC    | C0     | 00066                   | ADDL2                    |
|    |           |    | 52 |       | 5A    | D1     | 0006A                   | CPL                      |
|    |           |    |    |       | 1D    | 18     | 0006D                   | BGEQ                     |
|    |           |    |    |       | 02    | DD     | 0006F                   | PUSHL                    |
|    | 00000000G |    | 01 | FB    | 00071 | CALLS  | #1, DBG\$GET_TEMPMEM    | 3431                     |
|    |           |    | 67 | 50    | D0    | 00078  | MOVL                    | R0, (SUB_STRING)         |
|    |           |    | 57 | 67    | D0    | 0007B  | MOVL                    | (SUB_STRING), SUB_STRING |
|    |           |    | 59 | 01    | AA    | 9E     | 0007E                   | MOVAB                    |
|    |           |    | 52 |       | 59    | C3     | 00082                   | SUBL3                    |
| 50 |           |    | 56 | 01    | A0    | 9E     | 00086                   | MOVAB                    |
|    |           |    |    |       | 99    | 11     | 0008A                   | BRB                      |
|    |           |    |    |       | 67    | D4     | 0008C                   | 5\$:                     |
|    |           |    | 50 |       | 5B    | D0     | 0008E                   | 6\$:                     |
|    |           |    |    |       | 04    | 00091  | RET                     |                          |

; Routine Size: 146 bytes, Routine Base: DBG\$CODE + 1A42

: 3333 3447 1  
: 3334 3448 0 END ELUDOM

.EXTRN LIB\$SIGNAL, SYSSUNWIND

PSECT SUMMARY

| Name      | Bytes | Attributes                                                 |
|-----------|-------|------------------------------------------------------------|
| DBG\$OWN  | 12    | NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, PIC, ALIGN(2) |
| DBG\$CODE | 6868  | NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(0)   |
| DBG\$PLIT | 3257  | NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(0)   |

Library Statistics



| File                                     | -----<br>Total | Symbols<br>Loaded | -----<br>Percent | Pages<br>Mapped | Processing<br>Time |
|------------------------------------------|----------------|-------------------|------------------|-----------------|--------------------|
| -\$255\$DUA28:[SYSLIB]LIB.L32;1          | 18619          | 56                | 0                | 1000            | 00:01.8            |
| -\$255\$DUA28:[DEBUG.OBJ]STRUCDEF.L32;1  | 32             | 1                 | 3                | 7               | 00:00.1            |
| -\$255\$DUA28:[DEBUG.OBJ]DBGLIB.L32;1    | 1545           | 142               | 9                | 97              | 00:02.0            |
| -\$255\$DUA28:[DEBUG.OBJ]DSTRECRDS.L32;1 | 418            | 163               | 38               | 31              | 00:00.3            |
| -\$255\$DUA28:[DEBUG.OBJ]DBGMSG.L32;1    | 386            | 7                 | 1                | 22              | 00:00.3            |

# COMMAND QUALIFIERS

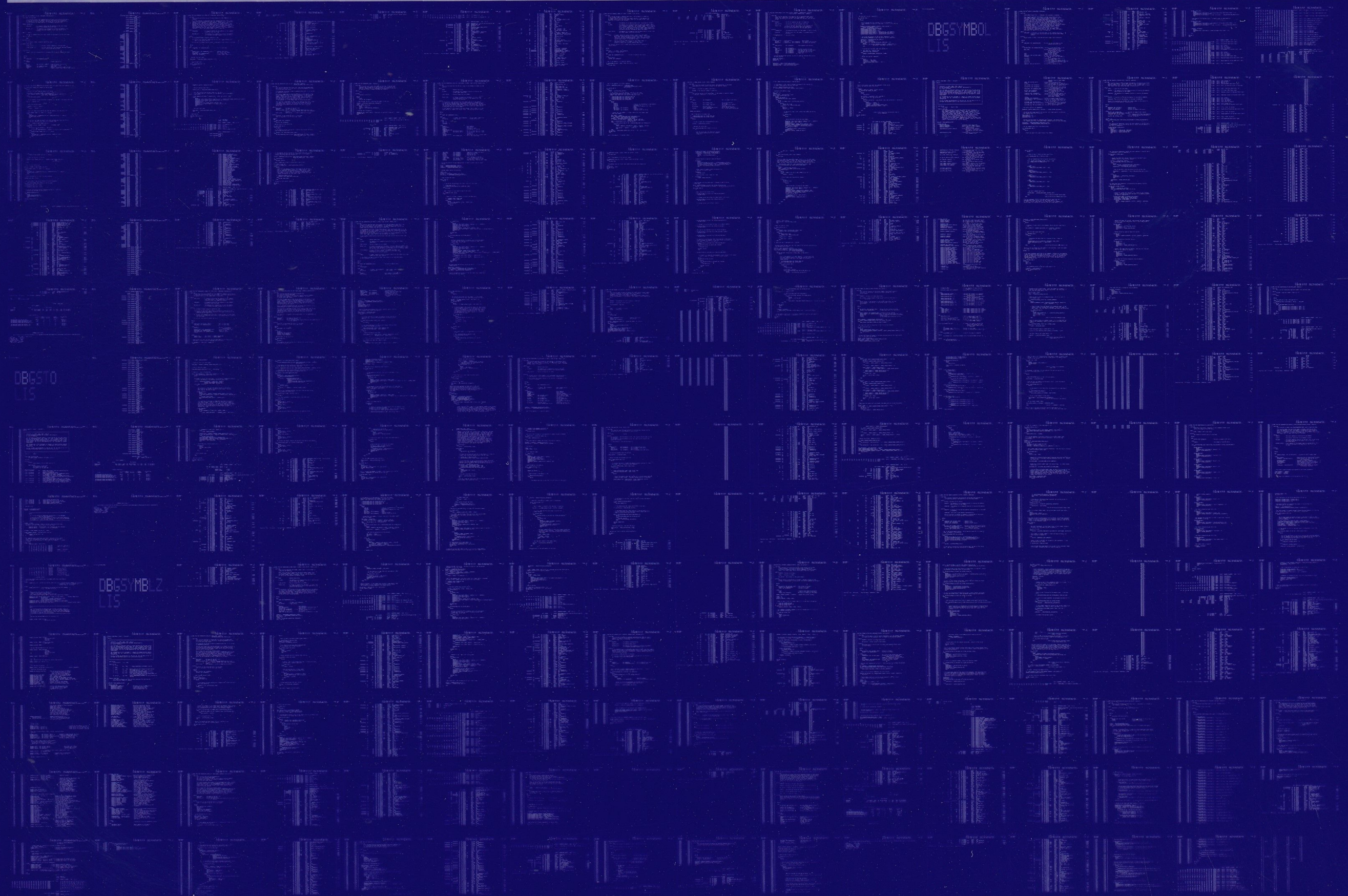
BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:DBGSYMBOL/OBJ=OBJ\$:DBGSYMBOL MSRC\$:DBGSYMBOL/UPDATE=(ENH\$:DBGSYMBOL)

Size: 6868 code + 3269 data bytes  
Run Time: 01:51.9  
Elapsed Time: 02:02.7  
Lines/CPU Min: 1849  
Lexemes/CPU-Min: 13529  
Memory Used: 561 pages  
Compilation Complete



0095 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY





0096

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY